Accepted for publication on ACM Journal on Data and Information Quality, 2023

Pipeline Design for Data Preparation for Social Media Analysis

CARLO A. BONO*, CINZIA CAPPIELLO*, BARBARA PERNICI*, EDOARDO RAMALLI*, and MONICA VITALI*, Dept. of Electronics, Information, and Bioengineering, Politecnico di Milano, Italy

In a data-driven culture, in which analytics applications are the main resources for supporting decision-making, the use of high-quality datasets is mandatory to minimize errors and risks. For this reason, data analysis tasks need to be preceded by a data preparation pipeline. The design of such a pipeline is not trivial: the data analyst must carefully choose the appropriate operations considering several aspects. This is often performed by adopting a trial-and-error approach that does not always lead to the most effective solution. In addition, extracting information from social media poses specific problems due to the need to consider only posts relevant for the analysis, for its dependence from the context being considered, for its multimedia contents, and for the risk of filtering out informative posts with automatic filters. In this paper, we propose a systematic approach to support the design of pipelines that are able to effectively extract a relevant dataset for the goal of the analysis of data from social media. We provide a conceptual model for designing and annotating the data preparation pipeline with quality and performance information, thus providing the data analyst preliminary information on the expected quality of the resulting dataset in a context-aware manner. The generation of metadata related to the processing tasks has been recognized as essential for enabling data sharing and reusability. To this aim, the dataset resulting from the pipeline application is automatically annotated with provenance metadata to get a detailed description of all the activities performed by the pipeline on them. As a case study, we consider the design of a pipeline for creating datasets of images extracted from social media in order to analyze behavioural aspects during COVID-19.

CCS Concepts: • **Information systems → Data management systems**; **Information integration**; **Decision support systems**; • **Human-centered computing**;

Additional Key Words and Phrases: Data preparation pipeline, Human-in-the-loop processes, Data provenance,

ACM Reference Format:
Carlo A. Bono, Cinzia Cappiello, Barbara Pernici, Edoardo Ramalli, and Monica Vitali. 2022. Pipeline Design for Data Preparation for Social Media Analysis. *ACM J. Data Inform. Quality* 1, 1 (April 2022), 24 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The analysis of data can be improved with the definition of a pipeline in which data lifecycle and processing steps are systematically defined. Several authors have discussed the different phases and tasks associated with data analytics pipelines [31]. At a higher level, a pipeline is composed of two main phases: (i) Data preparation and (ii) Data analysis. The former phase defines the data to be collected, and the preparation activities needed to transform them before their analysis. The

*All authors contributed equally to this research.

Authors' address: Carlo A. Bono, carlo.bono@polimi.it; Cinzia Cappiello, cinzia.cappiello@polimi.it; Barbara Pernici, barbara.pernici@polimi.it; Edoardo Ramalli, edoardo.ramalli@polimi.it; Monica Vitali, monica.vitali@polimi.it, Dept. of Electronics, Information, and Bioengineering, Politecnico di Milano, Piazza Leonardo da Vinci 32, Milan, Italy, 20133.

latter phase focuses on the specific analysis to be performed and the visualization and evaluation of the obtained results. In this paper, we focus on the data preparation pipeline that has the goal of guaranteeing the reliability and relevance of the data that will be analyzed. Appropriate data preparation is needed since real-world datasets are often “noisy”, i.e., they might contain errors, inconsistencies, or hardly relevant data that can negatively impact the results and their analysis. This is especially true for social media data: it has been shown that approximately only 5 out of 1,000 images from Twitter searches might be relevant in a specific case study [23]. This percentage may vary depending on the subject to be analyzed, but it is clear that data quality is an issue in analytical processes using social media sources.

To mitigate these problems, a pipeline is usually built with different kinds of components that manipulate the data in order to improve their quality and relevance before analysis. It is also essential to keep track of the tasks performed on data during the preparation process, as this provenance information could be useful for enabling the transparency and reuse of data in a data-sharing ecosystem. This is usually done without a systematic approach to the design of the pipeline and the generation of provenance information.

In this paper, we aim to support social media data analysts in the design of data preparation pipelines (hereinafter simply referred to as “pipelines”) with a systematic approach. We envision a human-in-the-loop approach in which the data analysts play a central role in every phase of the design process. First of all, they drive the definition of the dataset and data preparation pipeline configuration according to the goal of the analysis. The proposed approach supports the data analysts by executing the designed pipeline on a sample dataset, collecting information about the characteristics of the resulting refined dataset, and evaluating the contribution of each component (i.e., steps of the data preparation process) to the output dataset and its fitness for the goal. In this way, the approach is able to provide an estimation of the quality of the pipeline results and associate quality and provenance metadata with the resulting dataset. The data analyst can exploit the generated metadata for the redefinition and refinement of the pipeline’s components, constraints, and parameters through a feedback dashboard. In this way, the data analyst designs the final data preparation pipeline that will be executed on the full dataset to analyze. If the final results are not satisfactory or requirements change over time, a redesign could be required: in this case, the data analyst can simply reiterate the interactive design process.

The main contributions of this paper are the following:

- Definition of a human-in-the-loop approach for the design of data preparation pipelines with an iterative process for their configuration, validation, and execution.
- Classification and formalization of the data preparation components, considering functional and non-functional properties.
- Definition of the metadata to collect during the data preparation design phase for provenance, trustworthiness, and transparency requirements.

The paper is structured as follows. In Sect. 2, we describe the proposed data preparation pipeline design process in the context of social media analysis. In Sect. 3, we illustrate the proposed model for pipeline components and their associated quality and performance annotations. In Sect. 4 and in Sect. 5 we discuss the data preparation pipeline configuration and validation phases with a human-in-the-loop approach and how provenance and quality metadata are generated during this process. In Sect. 6, we discuss the pipeline execution and the generation of the metadata associated with the refined datasets obtained by executing the data preparation pipeline with a designed configuration. In Sect. 7, we describe some implementation details of the approach. In Sect. 8, we illustrate the use of the approach in a case study based on COVID-19-related images from tweets. In Sect. 9, we discuss related work and in Sect. 10 we draw conclusions and discuss future work.

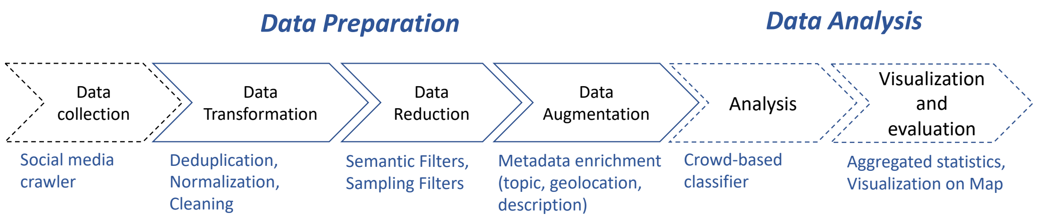
Data Preparation
Data Analysis
Data collection
Data Transformation
Data Reduction
Data Augmentation
Analysis
Visualization and evaluation
Social media crawler
Deduplication, Normalization, Cleaning
Semantic Filters, Sampling Filters
Metadata enrichment (topic, geolocation, description)
Crowd-based classifier
Aggregated statistics, Visualization on Map

Fig. 1. Example of social media analysis pipeline

2 SUPPORTING DATA PREPARATION FOR SOCIAL MEDIA ANALYSIS

2.1 Challenges of social media analysis

The goal of this paper is to support the design of production-ready data preparation pipelines that combine heterogeneous functionalities in order to refine datasets relevant to a given analysis. A typical pipeline is shown in Fig. 1, where the data retrieved are at first subject to a Data Preparation process and then used in a Data Analysis process. The Data Preparation phase is usually multi-step and mostly automated, while the Data Analysis phase usually requires a human-based analysis. Nevertheless, the design and configuration of a data preparation pipeline might require a relevant effort and several iterations.

Data preparation answers the need to improve the quality of the original datasets in many scenarios. When information is obtained from social media, extracting relevant information from these datasets can be challenging [17]. Valuable information is usually found among many other posts, such as comments, opinions, support messages, and unrelated messages. Tackling the distinctive volume, velocity, and variability of social media calls for automated techniques that focus on different aspects of understanding data, in order to control their quality. Moreover, the prevailing mediality of social media data requires multiple, tailored, and automated AI-based tools, rather than generalist or rule-based approaches. Most often, data quality is controlled by combining these tools and human expertise, in order to validate the information extracted, for example, by assigning tasks on crowdsourcing platforms [2]. In social media analysis, different application scenarios may have different constraints, as discussed in [2]. For real-time analysis, requirements are posed mainly on response time and acceptable accuracy estimations for prepared data, while in curated datasets with crowdsourced labeling, constraints refer to available human resources and the difficulty of the labeling tasks to be performed (e.g., validating geolocation). As a motivating example, social media can provide timely and first-hand information for real-time analysis of emergencies, as described for instance in [27] for earthquakes reporting, or descriptive analysis related to trending topics useful for crisis management (e.g., analysis of tweets during COVID-19 [23]). In these scenarios, the issues of data quality and timeliness are particularly relevant.

2.2 Proposed approach for data preparation pipelines design

Given a problem to be analysed, an input dataset to be prepared for the analysis (e.g., streaming data or batches of data obtained from one or more data sources), and a set of available tools to process it, the *Data Preparation Pipeline Design* (Fig. 2) consists in the identification of the most suitable sequence of steps to be performed, the configuration of the selected tools, and an assessment of the expected quality of the outcome of the preparation.

The *Data Preparation Pipeline Design* is driven by the *Goal* that the data analyst wants to achieve. The goal is defined before starting the pipeline design and it will drive the successive data preparation steps. Note that the problem of formulating such a goal is out of the scope of this paper.

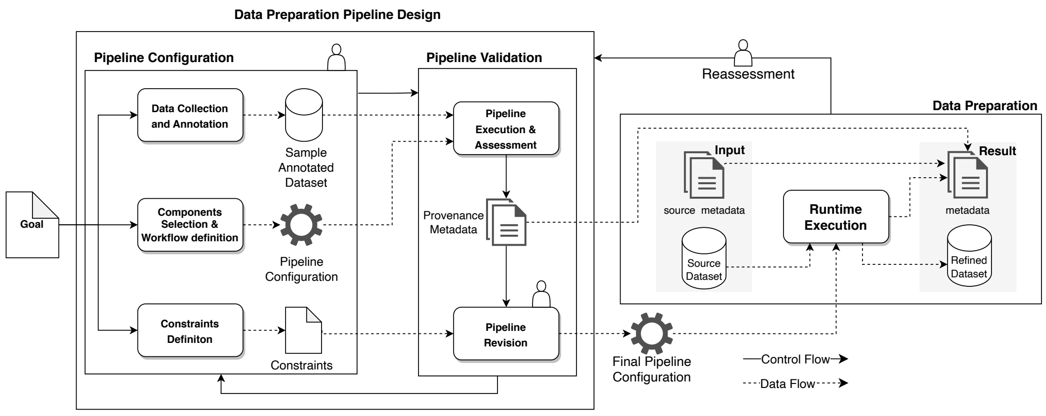

Fig. 2. Data preparation pipeline design process

The Data Preparation Pipeline Design is composed of two main phases: (i) Pipeline Configuration and (ii) Pipeline Validation.

In the *Pipeline Configuration phase*, the data analyst performs the following actions:

- *Data Collection and Annotation*: a representative sample is extracted from the initial dataset. In the annotation step, the annotators specify the relevance of the data items with respect to the predefined goal.
- *Components Selection and Workflow definition*: the data analyst selects, from the *Component Library* that is defined and modeled according to the component model described in Sect. 3, the components that have to be included in the pipeline and their execution order.
- *Constraints Definition*: data analysts can express some non-functional constraints related to performance and quality measures of the entire pipeline, e.g., an acceptable volume of output items, processing time and cost constraints, desired precision, and recall of the output.

The pipeline configuration phase provides three artifacts: (i) a *Sample Annotated Dataset*, (ii) a *Pipeline Configuration*, and (ii) the list of non-functional *Constraints*. These objects feed the Pipeline Validation phase, which aims to evaluate the efficiency and effectiveness of the initial pipeline configuration and improve it, if necessary.

The *Pipeline Validation phase* is performed in two steps:

- *Pipeline Execution and Assessment*: the pipeline is executed on the sample dataset, and the quality of the result is assessed with respect to the sample annotations. Execution information, including involved components and related configurations, and components and pipeline performances, are stored as *Provenance Metadata*.
- *Pipeline Revision*: the results of the pipeline execution in terms of performance and quality are presented to the data analyst through a feedback dashboard. In case of unsatisfactory outcomes, the data analyst might decide to modify the pipeline going back to the Pipeline Configuration phase. The system can also support the data analyst by suggesting enhancement actions such as component substitution and/or reconfiguration.

When the data analyst is satisfied with the pipeline configuration, the *Data Preparation* phase can be executed on the *Source Dataset*, i.e., the entire data source from which the sample set has been extracted. The data preparation pipeline takes as input such data together with their metadata in order to obtain the *Refined Dataset* that will be used in the data analysis. At the end of this phase, or even at a later moment, it might happen that the data analyst is not satisfied with the obtained results. This could happen for different reasons, e.g, the Source Dataset was not accurately represented by the sample data, and/or the characteristics of data changed over time. In this case, the redesign of the pipeline is needed and the whole process is reiterated.

The refined dataset generated through the pipeline can be used in a specific analysis but could be also published in a data ecosystem. In order to enhance the reusability of the dataset and the transparency of the data preparation phase, the refined dataset is enriched with metadata. Three types of metadata are associated with it (Sect. 6): (i) **source metadata**, directly obtained by the data source and associated with the source dataset, (ii) **execution metadata**, collecting information about the pipeline execution on the items of the source dataset (see Sect. 3); (iii) **provenance metadata**, generated in the configuration phase and capturing the pipeline characteristics. The availability of such metadata makes our approach also beneficial for the creation of data spaces in which the trustworthiness among partners is a key factor based on ensuring transparency in data preparation and high-quality of the shared data. For example, in data lakes, there is usually a raw data area that contains ingested data and a revised data area in which clean data sets are stored [16]. The proposed approach can be used to generate clean datasets from the raw data. Moreover, the availability of provenance metadata associated with the refined dataset allows data analysts to understand better if the data preparation tasks performed are suitable for their analysis.

3 COMPONENTS: DEFINITIONS AND PROPERTIES

In this section, we introduce the Components Library used in the Components Selection and Workflow Definition phase. Each step of the data preparation pipeline consists of the execution of a component able to operate some specific transformation on the dataset. In the proposed approach, we assume to provide the data analyst with a predefined library of components that can be selected and enacted. Each component is described through its behaviour, both in terms of functional and non-functional properties. This information can be exploited by the data analysts to select the combination of components that fit their purposes. In Sect. 3.1 we provide the necessary formal definitions, while in Sect. 3.2, we provide a model for the classification and formalization of the characteristics of the components of the library, and in Sect. 3.3 we define a model for the annotations with which the components are enriched in the proposed approach.

3.1 Component semantics

A generic component can be defined as in Def. 3.1.

Definition 3.1. A **Data Preparation Component** $c_i \in C$ is a self-contained module that performs a specific task, transforming an input dataset I into an output dataset I' by applying a set of operations that affect its data items and/or their metadata. C is the Components Library, the set of all available components. A component is defined as a tuple:

$$c_i = < n_i, d_i, IN_i, OUT_i, A_i > \tag{1}$$

where: n_i is the name of the component, d_i is the description of the component, IN_i (resp., OUT_i) is the type of input data (resp., output data), A_i is the set of annotations describing the non-functional characteristics of the component (e.g., estimated cost, time, and quality, as detailed in Sect. 3.3).

The component name n_i maps to an implementation that is responsible for processing the data items according to the description of its expected behaviour d_i. There is no formal guarantee that d_i matches the user needs; in our vision, the component description serves as a "good faith" contract for the functional properties of a component.

Component reification. When a component is selected for execution, a set of configuration parameters $Conf$ is provided. The configuration parameters are specific to each component and are expressed as key/value pairs. Typical configuration parameters are the input field name and confidence thresholds. An instantiated component is then defined as:

$$c_{i,Conf} = <c_i, Conf> \tag{2}$$

We formally describe component instances as defined in Eq. 2 using an extended Backus–Naur form[1] as a metasyntax notation. Since this syntax covers a subset of the JavaScript Object Notation (JSON) expressiveness, it seems natural to use JSON to manage component instances at an operational level, e.g., for pipeline configuration and logging.

Data and execution metadata. A dataset on which a component operates is defined as $I = \{D_I, M_I\}$, where D_I are the data items and M_I are the associated metadata.

Definition 3.2. D_I is an instance of a relational schema R_{D_I}:

$$R_{D_I}(X_1 : T_1, X_2 : T_2, \ldots, X_n : T_n) \tag{3}$$

where X_i denotes the attribute name and T_i denotes the data type. For example, in the context of social media, a data item corresponds to a post, composed of typed data fields such as the post and the author identifiers, the textual contents, the URLs of the included images and so on.

When a component is executed on a data item, it usually appends metadata to the item. For example, quantitative measures regarding the execution of a component on each data item are tracked, such as processing time or output confidence (see Sect. 5.2 for a comprehensive overview of the methodology). Execution metadata can be seen as an instance of a relational schema $R_{M_{I,Exec}}$ as well, where also the generating component instances are associated:

$$R_{M_{I,Exec}}(Y_1 : (T_1, c_{1,Conf}), Y_2 : (T_2, c_{2,Conf}), \ldots, Y_n : (T_n, c_{n,Conf})) \tag{4}$$

Elements $m \in M_{I,Exec}$ have a one-to-one mapping with elements $d \in D_I$. More generally, when a component is executed on a data item, it usually produces a set of metadata so that $R_{M_{I'}} \supset R_{M_I}$.

3.2 Classification of data preparation pipeline components

Selecting a proper combination of components for a specific dataset and a specific goal of the analysis is not trivial even for expert users. Most of the time several attempts are needed before finding an effective pipeline. The Components Library makes this activity easier by providing a rich, organised set of components, enriched with a description of their capabilities and a set of metadata about their performance. We can classify the library components according to three main categories [10]: (i) Data reduction, (ii) Data transformation, and (iii) Data augmentation components.

Data reduction components (filters). Produce an output dataset I', where $D_{I'} \subseteq D_I$ and $R_{D_{I'}} = R_{D_I}$. We focus on two main types of such components: semantic and sampling filters. *Semantic filters* in our context reduce the number of data items, i.e., photos on the basis of a specific selection configuration $K \equiv Conf$: $I' = f_{filter}(I, K)$ where K can be related to the content of the photo or to metadata that describes the photo. In the former case, the data analyst can specify K as the presence of an element (e.g., food, a given number of persons), the specification of the environment (i.e., indoor vs. outdoor), or the identification of a scene (e.g., cafeteria, theater, playground). K can also refer to non-media attributes, such as conditions over the posting time, the availability of geolocation data, the popularity of the post (e.g., the number of likes), or the reliability of the author of a post. *Sampling filters* reduce the number of data items within a certain reduction rate that expresses the proportion of filtered-out data. For example, a sampling filter could use a random

[1]The specification can be found at https://bit.ly/30CNVcU

sampling technique: elements of I' are selected randomly from I. Sampling could also be performed using different approaches, such as systematic sampling or clustering sampling [19].

Data transformation components. Transform data items in I yielding $D' = f_{transf}(D)$, with $R_{D'_I} \supset R_{D_I}$ if the transformed data are appended to the original data, or $R_{D'_I} = R_{D_I}$ if the transformed data replace the original data. They include *Normalization* and *Cleaning tasks*. The former refers to the process of transforming data into a standard form in a way to guarantee a certain representation consistency. The latter aim to identify and correct errors and anomalies (e.g., anomaly detection, missing values imputation), resolve conflicts, and alter inappropriate or useless data.

Data augmentation components. Produce an output dataset I', where $R_{D'_I} \supset R_{D_I}$. These components generate additional data (e.g., location, topic, image description) that can be extracted from the available information (e.g., infer the location of a tweet from the text) or derived from a more complex analysis.

3.3 Enriching components with quality and performance annotations

The components presented in Sect. 3.2 are enriched with annotations A_i that can be exploited by the data analysts to select the set of components that better fit their goals. Annotations can capture several aspects related to the component behaviour. Without losing in generality, here we focus on three main aspects: (i) the time required by the component to perform its function, (ii) the cost of the resources required to perform the task, and (iii) the quality of the output provided by the component. We define how these aspects are measured for each component.

Time. Social media analysis pipelines are often used to support the data analyst in taking timely decisions in the context of emergency management. The time required for processing the items to be analysed might affect the responsiveness of the decisions of the data analyst. Each component is characterised by a different execution time, and the overall execution of a data preparation pipeline might be also affected by the order in which the components are executed.

Definition 3.3. The **Time** required by a component $c_i \in C$ to process a single item in the data set is indicated as $time_{c_i}$. The time required to process a dataset I composed of L data items can be expressed as a function: $time_{c_i}(L)$, where $time_{c_i}(1) = time_{c_i}$. In case the component is characterised by a linear behaviour, $time_{c_i}(L) = time_{c_i}(1) \times L$.

Cost. The processing activity is likely subject to budget constraints. The cost for each component depends on the amount of resources required for its execution. The following definition holds for both automatic (e.g., ML models) and manual (e.g., crowd) components:

Definition 3.4. The **Cost** required to process a single data item in the dataset with a component c is indicated as $cost_{c_i}$. It is estimated by monitoring the amount of resources employed to process a single item. The cost required to process a dataset I composed of L data items can be expressed as a function: $cost_{c_i}(L)$, where $cost_{c_i}(1) = cost_{c_i}$. In case the component is characterised by a linear behaviour, $cost_{c_i}(L) = cost_{c_i}(1) \times L$.

Quality. Quality annotations capture the effectiveness of the component in achieving its purpose. To express quality, we selected the following set of metrics, mainly useful for Data Reduction components. However, additional quality metrics can be supported.

Definition 3.5. **Precision** $precision_{c_i} \in [0, 1]$ assesses the ability of the component to discriminate items that are not relevant to the task. Considering an input dataset I with u relevant items and an output dataset I' with v relevant items, $precision_{c_i} = \frac{v}{|I'|}$

Definition 3.6. **Recall** $recall_{c_i} \in [0, 1]$ assesses the ability of the component to retain items that are relevant to the task. Considering an input dataset I with u relevant items and an output dataset I' with v relevant items, $recall_{c_i} = \frac{v}{u}$

Definition 3.7. **Representativeness** $representativeness_{c_i}$ ensures that the sets I and I' maintain the same distribution of values. Considering an input set I representing H distinct values (or type of images in our example) and an output set I' representing $J \subset K$ distinct values with $|I_h|$ the number of occurrences for the value $h \in H$ and $|I_h| > q$, where q is the minimum allowed number of occurrences per value can be measured as $representativeness_{c_i} = \frac{|J|}{|H|}$.

Definition 3.8. **Reduction Rate** $reduction_{c_i} \in [0, 1]$ assesses the ratio of discarded elements. Considering an input set I and an output set I', $reduction_{c_i} = 1 - \frac{|I'|}{|I|}$.

To obtain the annotations A_i for each component, two different approaches are combined: (i) Nominal annotations and (ii) Context-dependent annotations. Nominal annotations express the nominal performance of the component and are provided in the Component Library. These annotations are obtained by testing the component behaviour on a set of reference datasets in the state of the art, aggregating the results. For instance, a component in charge of detecting the presence of objects in an image will be tested on several image datasets to validate its behaviour. The annotation process is executed on relevant datasets when the component is added to the Component Library and can be updated over time with new executions on new datasets.

Nominal annotations are not specific for the data analyst's purposes but can be used to have preliminary knowledge about the component behaviour. Instead, context-dependent annotations are generated when the component is selected as part of a pipeline, and reflect performance measures over an application-specific dataset. Context-dependent annotations will be discussed in Sect. 5.

4 PIPELINE CONFIGURATION PHASE: METHODS AND TECHNIQUES

In this section, we describe the steps for the configuration of a data preparation pipeline. In this phase, the proposed methodology aims to support the data analyst by providing a structured pipeline configuration process, supported with relevant metadata. In particular, as depicted in Fig. 2, the pipeline configuration consists of three tasks, whose order of execution is not relevant: (i) Data Collection and Annotation, (ii) Components selection and Workflow definition, and (iii) Constraints definition.

4.1 Data Collection and Annotation

The data analyst needs to define the source of the data. Social media data sources are characterised by a high and variable in time volume of items. Also, these data sources are not static, and after the definition of the pipeline items continues to be generated. The aim of the data analyst is to extract from these data sources items relevant to the analysis phase.

Definition 4.1. We model a source description $\mathcal{SD}$ as the set of parameters needed to retrieve the data items from a social media data provider. A source description can be defined as $\mathcal{SD} :< G, Q, T >$, where G is the social media from which the data items are going to be extracted (e.g., Twitter, Flickr), Q is the query expressed as a set of keywords used for selecting items related to the topic of interest, and $T =< T_{start}, T^*_{end} >$ is the time interval in which the items to be selected have to be published. The value of T^*_{end} is optional.

The defined source description is used as input for a crawler that collects the data items from the selected social media. The output of the crawler is the set of data items used as input for the pipeline. If T^*_{end} is not specified, the crawler keeps collecting the new data items that are added on social media to enrich the original data items.

Given the provided definitions, we can now define the Source dataset $I_O = \{D_O, M_O\}$ as:

$$I_O = crawl(\mathcal{SD})$$

where the subscript O denotes the original dataset, $crawl$ is the function in charge of crawling data items from a social media data source, D_O is the set of items collected by the crawler, M_O are the metadata associated with this dataset, and $\mathcal{SD}$ is a source description as specified in Def. 4.1.

In order to enable the successive steps of the methodology, part of the original data items needs to be annotated (*Sample Annotated Dataset*), providing information about their relevance for the analysis. We thus define $\hat{I_O} = \{\hat{D_O}, \hat{M_O}\} \subseteq I_O$ as a sample of the dataset I_O selected for the annotation process. In this paper, we are not going into details on how the size and content of $\hat{I}_O$ are determined, but the reader can refer to several studies focusing on this specific challenge [19].

The annotation process for the Sample Annotated Dataset $\hat{I}_O$ can be performed by crowd workers, providing them with relevant and detailed information on the purpose of the analysis. The collected annotations are used to evaluate the ability of the pipeline to keep relevant items while discarding useless ones. This process will be explained in more detail in Sect. 5.

4.2 Components Selection and Workflow Definition

During the Pipeline Configuration, a set of components are selected from the Components Library. As discussed in Sect. 3, the Components Library lists the set of available components together with the information about their functional and non-functional characteristics. The data analyst exploits the provided description to select a proper set of components. As an example, if the goal of the analysis is to understand if people are wearing or not masks in public places, the pipeline will require a component able to detect the presence of an object (e.g., a person) in a picture and a component to detect if the location is a public or a private place.

The selected components need to be instantiated in order to be used. This requires the definition of the values of the configuration parameters $Conf_i$ available for each component (see Def. 3.1).

In this step, the data analyst also defines the order in which the components will be applied, composing a workflow. A pipeline consists of a combination of components executed following a specific workflow.

Definition 4.2. A data preparation **Pipeline** $\mathcal{P}$ is defined as an ordered set $C = (c_i)_{i=1}^{K}$ of components, where c_i is a single component and K is the total number of components selected. A component is an elementary work unit used to process a dataset I in input to generate a new dataset I' in output. The **Pipeline** is enriched with a set of annotations A_P, describing its non-functional characteristics. Thus, a **Pipeline** is defined as $\mathcal{P} =< C, A_P >$.

For simplicity, in this paper, we consider only sequential pipelines in which the components are executed in series (each component c_i is preceded by the component c_{i-1} and followed by the component c_{i+1}). However, the approach could be generalized to more complex pipelines, defining data analysis workflows in which some components are executed in parallel or replicated.

To support data analysts during this phase and to give them tools to understand if the designed pipeline has satisfactory non-functional properties, the pipeline annotations A_P are estimated starting from the annotations of the selected components. The **time** and **cost** required to execute a pipeline can be estimated as:

$$time_P = \sum_{c_i \in Pipeline} time_{c_i}(L_{c_i}) \qquad cost_P = \sum_{c_i \in Pipeline} cost_{c_i}(L_{c_i}) \tag{5}$$

where L_{c_i} is the number of items expected in input for each component c of the pipeline. This number will depend on the order of the components and on their reduction rate. As a consequence, the order of components influences the total time and cost of the pipeline. The number of items in output of the whole pipeline will influence the cost for the following analysis tasks. When humans are involved in the analysis through crowdsourcing activities, the task owner is in charge of defining the price per task, owed to each crowd-worker. It is worth noticing that the price set might affect the quality of the result, as discussed in [21], and consequently selecting a proper reward is essential for a successful outcome of the analysis tasks. In this paper, we assume the price per task is given and known at design time.

Reduction rate, measured at the pipeline level, can be estimated as:

$$reduction_P = 1 - \prod_{c_i \in Pipeline} (1 - reduction_{c_i}) \tag{6}$$

The **number of items** submitted to a component L_{c_i} can be obtained as:

$$L_{c_i} = (1 - reduction_k) \times |D_I| \tag{7}$$

where $reduction_k$ is the overall reduction rate at the k-th step of the pipeline and $|D_I|$ is the number of items of the original dataset.

The estimation of the overall **precision** of a pipeline $precision_P$ is not trivial given that the precision of a component can affect the dataset in input of the following ones. The overall pipeline can be enriched with precision related metrics at the pipeline level like $precision\_max_P$, $precision\_min_P$, and $precision\_avg_P$, defined as the maximum, minimum, and average precision of the components composing the pipeline. Similarly, **recall** related metrics at the pipeline level are $recall\_max_P$, $recall\_min_P$, and $recall\_avg_P$.

At the end of the pipeline, we expect the precision and recall measures to be defined by the following intervals:

$$precision_P \in \left[\prod_{c_i \in Pipeline} precision_{c_i} , precision\_min_P \right] \tag{8}$$

$$recall_P \in \left[\prod_{c_i \in Pipeline} recall_{c_i} , recall\_min_P \right] \tag{9}$$

In the lower bound, we assume independence between the output of the components. These equivalences hold when the behaviour of the pipeline is equivalent to the logical conjunction of its K components. In the upper bound, we instead consider the case in which the components have relevant overlap in the set of items in output, which is likely when they are defined to reach a common goal as in the pipeline. These boundaries apply to common scenarios, while precision and recall might have values beyond them in extreme cases (i.e., the components are not suitable for the goal or are redundant).

4.3 Constraints definition

In the Pipeline Configuration phase, the data analyst also defines the desired non-functional properties (i.e., performance dimensions). Non-functional properties are based on the time, cost, and quality aspects introduced in Sect. 3.3 and are expressed as constraints.

Definition 4.3. A pipeline $\mathcal{P}$ is associated with a set $\mathcal{R}$ of constraints. A **constraint** $r \in \mathcal{R}$ expresses a desired behaviour measurable through a non-functional property. It can be defined as a tuple $r =< e, v, p >$, where e is the property (i.e., cost, time, or quality), v is a reference value for the criteria, and p is a comparison operator.

Constraints can thus be expressed on the time needed to process a dataset through the whole pipeline, on the cost of the execution, or on quality metrics such as precision, recall, volume, and representativeness. An example of constraint could be "$precision > 0.9$". All the constraints in $\mathcal{R}$ need to be satisfied at the same time (AND relation of all $r \in \mathcal{R}$). Such constraints should be satisfied within the execution of the pipeline on the entire dataset. Note that during the pipeline validation phase in which the pipeline is executed on the sample annotated dataset, the constraints related to time and cost will be derived on the basis of the size of the sample dataset.

In Sect. 5, we will discuss how the pipeline designed in the configuration phase is enriched with quality, time, and cost annotations, which can be used to verify the satisfaction of the constraints defined by the data analyst.

5 PIPELINE VALIDATION PHASE: METHODS AND TECHNIQUES

Once the pipeline configuration is completed, it must be validated to check its ability to fulfill the purposes of the data analyst according to the goal. The validation is based on the execution of the preparation pipeline on the Sample Annotated Dataset $\hat{D}_O$ and based on a human-in-the-loop revision of the results performed by the data analyst to improve data preparation.

In the pipeline validation phase of our methodology, the following main steps are performed:

(1) *Pipeline Execution and Assessment* on the Sample Annotated Dataset (Sect. 5.1).
(2) *Pipeline Revision* (Sect. 5.3).

In this section, we illustrate these steps toward reaching the preferred configuration for the data analyst. Both tasks see the use of a Provenance Model (Sect. 5.2), exploited to keep track of the evolution of the pipeline configuration: firstly, it is used to keep track of the performance of a pipeline configuration; in addition, this information is used by the data analyst as a basis for monitoring and refining the configuration itself if needed.

5.1 Pipeline Execution and Assessment

The Pipeline Execution and Assessment step is an automated activity that is executed to validate the pipeline configured in the Pipeline Configuration phase using the Sample Annotated Dataset.

In this phase, the designed data preparation pipeline is executed on the sample dataset, using the designed pipeline configuration. Evaluation of data sampled from a specific real-world scenario is critical since performance measures can be highly variable depending on the distribution of input data. Both quality and performance are evaluated for each of the selected components and for the whole pipeline execution.

Considering the single components, it is possible to complete the nominal annotations present in the Component Library with context-dependent annotations, as anticipated in Sect. 3.3. Context-dependent annotations measure the fitness of the component for a specific purpose. These annotations are obtained by executing the set of selected components using the Sample Annotated Dataset as input. The output of the execution of each component is a dataset I_C which is compared with the labels provided in $\hat{I}_O$. From this comparison it is possible to compute, for instance, the precision and recall metrics for the component, as well as the reduction rate computed as $1 - \frac{|D_C|}{|\hat{D}_O|}$, where D_C is the set of items in I_C and $\hat{D}_O$ is the set of items in $\hat{I}_O$. During the execution, it is also possible to keep track of the time and cost of execution in the metadata M_C. All this information is used to verify the fitness of the selected components for the data analyst goal.

Similarly, also the whole pipeline P is executed using the sample dataset $\hat{I}_O$ as input, generating the result dataset I_P as output. Also in this case, precision, recall, and reduction rate $(1 - \frac{|D_P|}{|\hat{D}_O|})$ can be computed based on the sample data annotations, and time and cost can be measured.

Both single component annotations and pipeline annotations will generate Provenance Metadata for the pipeline execution, recorded according to the provenance data model described in Sect. 5.2. This information will be used by the data analyst to revise the pipeline configuration as described below in Sect. 5.3.

5.2 A provenance model for data preparation pipelines

Data provenance, or data lineage, consists of facts related to a piece of data that support the quality and reliability of the data itself. To this aim, provenance is a collection of metadata that contains information about entities, activities, and people involved in producing data, answering questions such as when, how, why, where, and by whom a dataset was created. Provenance is helpful to replicate results, track down errors or meticulously report the whole data creation process guaranteeing data transparency. Finally, the provenance data model defines the structure (the schema) of the provenance metadata for a particular case study by relating the various entities, activities, and people involved in the data creation.

In this work, as in all user-driven data preparation approaches, provenance has a central role when data are manipulated by a series of pipeline steps for preparation. For example, in a pipeline that manages scientific data to develop a data-driven model, it is hard to disambiguate which phase or procedure of a long and complex process is responsible for the improvement or deterioration of the model. Provenance, keeping track of each action on the data and on the model, can help in this task and make it possible to replicate scientific results [11]. In general, the data analyst can be motivated to use provenance for multiple purposes such as accountability, reproducibility, or process debugging [20], but in our methodology, it also increases transparency and trustworthiness.

We adopt the W3C PROV data model to design the provenance data model paying attention to the data sheets directives [14] representing what is strictly necessary for the data preparation pipeline design. In fact, a provenance data model can have different levels of granularity [20], and thus different verbosity: we use a combination of the data and workflow level of details such that a future user can adequately reconstruct the dataset generation and to keep track of each modification of the pipeline configuration during the enhancement iterations of our methodology.

The W3C PROV data model is a conceptual provenance model built around three concepts: *Entity*, *Activity*, and *Agent* [4]. Briefly, an *Entity* is something for which we want to trace the provenance. An *Activity* is an operation performed on an entity to produce another entity or another version of it. Finally, the *Agent* is something or someone that bears the responsibility for an action or an entity.

The PROV data model is adopted to specifically track the provenance in data preparation pipelines for social media data as shown in Fig. 3. Since it must fully track the evolution of the pipeline configuration and it also must enable the reproducibility of the refined dataset generation, this provenance model is transversal to the pipeline configuration and validation phases presented in Sect. 2.

The PROV data model records metadata about the Sample Dataset extraction, the Pipeline configuration, and the execution of the pipeline with a given configuration, as well as the sequence of revisions of the configuration performed by the data analyst.

Following our case study, but without losing generality, a *Sample Dataset* is generated by a *Crawling* action using a specific *Crawler* on a given *Source*. For social media analysis, it is important to keep track of the search parameters such as the query and the time interval, which are associated with the Crawling action, and when the *Crawler* was interrogated to collect data. Even the type of used *Crawler* could have an impact on the final results. For example, Twitter, which could be a *Source* of social media data, has different types of API (developer, academy, business, etc.), each with a different level of granularity in collecting tweets and thus different final results. For this

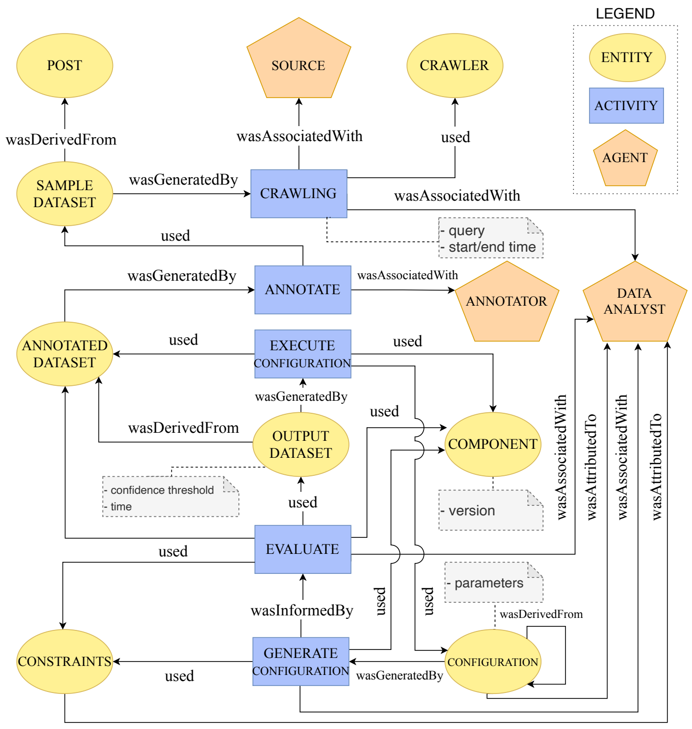

Fig. 3. The conceptual provenance data model for social media data preparation pipelines

reason, inside the *Crawler* agent, it is important to specify the crawler version. The resulting *Sample Dataset* is a set of *Posts*. Once the *Sample Dataset* is available, the *Annotator* is responsible for the execution of the *Annotate* action on the collected data, generating the corresponding *Annotated Dataset*. The *Data Analyst* generates also the first initial *Configuration* of the pipeline with the *Generate Configuration* action. A *Configuration* includes *Components*, their parameters, and the Components' workflow. The *Constraints* defined by the *Data Analyst* for the overall pipeline are recorded. *Execute Configuration* records the *Component* actions using a given *Configuration* to produce the resulting *Output Dataset*; for each *Component* its characteristics are recorded (e.g., component version). The time and cost of the pipeline execution with a given configuration are recorded in *Execute Configuration*. The *Evaluate* action by the *Data Analyst* is based on the assessment of the *Output Dataset* against the *Annotated Dataset* and *Constraints*. Leveraging the quality performance information of the overall pipeline and of each *Component*, the *Data Analyst* may decide to generate (*Generate Configuration*) a new *Configuration* as an improvement of the previous one (*DerivedFrom* on *Configuration*) or modify the *Constraints* to reach the desired outcome. The actions *Execute*, *Evaluate*, and *Generate* may be iterated until the *Constraints* specified for the pipeline are fulfilled and the *Data Analyst* confirms the final *Configuration*. The whole process is recorded as metadata associated with the prepared dataset, as it describes how the *Configuration* for data preparation was achieved. Hence, they store quality, cost, and time information about the pipeline execution.

5.3 Pipeline Revision

The information generated during the Pipeline Execution and Assessment, including the observed behaviour of the pipeline and of each component on the Sample Annotated Dataset $\hat{D}_O$, is used to generate a feedback dashboard that is presented to the data analyst. The feedback dashboard aims to provide insights on the fitness of the pipeline to the goals of the data analyst, as well as suggest improvements in the pipeline design and configuration.

More specifically, the feedback dashboard consists of the following parts:

- **Insights on the behaviour of single components.** As we discussed in Sect. 5.1, each component of the pipeline is executed on the Sample Annotated Dataset and the obtained result is compared with the labels provided during the Data Collection and Annotation activity. This information enables the computation of contextual annotations on the component, in terms of quality, time, and cost. However, the plain execution gives only results dependent on the selected parameters set for the component execution, which can be sub-optimal. In order to support the data analyst in identifying the best value for the parameters, a grid search parameter exploration is applied in this phase, simulating the execution of the component with different values of the parameters (e.g., the component's confidence threshold) and collecting data on how this affects the considered constraints (e.g., precision, recall, and reduction rate). The resulting information is included in the dashboard and presented to the data analyst in the form of a graph. Interacting with the dashboard, the data analysts can explore the trade-off in terms of quality and efficiency for the different values of the parameters and select the best configuration according to their needs.
- **Insights on the order of the components in the pipeline.** In Def. 4.2, we have defined the pipeline as an ordered set of components. The order in which the components are executed can affect the constraints' satisfaction. In fact, different components are characterised by different quality, different execution times, and different reduction rates. Thus, the order of the components has to be taken into account to obtain an effective and efficient pipeline. Once again, it is difficult for the data analyst to know in advance which is the best order given a set of components. In this phase, the characteristics of each component in terms of execution time and reduction rate are considered and a heuristic ordering is suggested based on the observation that: (i) components with higher reduction rates should be executed at the beginning of the pipeline to limit the number of items to be analysed by the other components; (ii) more time-consuming components should be located at the end of the pipeline, where most of the unwanted items have already been filtered;
- **Insights on the behaviour of the pipeline.** Comparing the dataset generated by the pipeline execution with the annotated dataset, it is possible to compute time, cost, and quality values for the whole pipeline and compare them with the constraints verifying their satisfaction. Even for the whole pipeline, starting from the configuration selected by the data analyst, we execute a grid exploration of the combination of the parameters' values and build a graph showing the effect of the different parameter values on the pipeline configuration constraints selected by the data analyst. This exploration highlights an approximation of the Pareto frontier for the configuration choices, as well as admissible regions for the configuration, i.e., configurations that fulfill the constraints.

Exploiting the feedback dashboard, if needed, the data analyst can activate a feedback loop by changing the pipeline configuration (e.g., adding/removing components, changing configuration parameters, redefining constraints, changing the order of the components). The refined pipeline will be subject to a new validation phase and this process will be iterated until the data analyst is satisfied with the results and a final pipeline configuration is obtained.

6 PIPELINE RUNTIME EXECUTION AND ADAPTATION

Once the data analyst is satisfied with the obtained configuration, the resulting pipeline configuration is ready to be used to prepare the dataset (or a series of datasets with similar characteristics), including provenance metadata.

With reference to Fig. 2, we assume to have a *Source dataset* to be prepared, including its own *Source metadata*, providing information on their provenance. The *Source metadata* can follow the structure of the PROV model (Fig. 3) for the *Crawling* action if the dataset is the result of a crawl on a social media, or it can follow another format (e.g., datasheets for annotating datasets [14]); it could also be the result of a previous application of the PROV model illustrated in Sect. 3.

The Source dataset is prepared to execute the data preparation pipeline with the *Final pipeline configuration* and its *Result* is an output consisting of a *Refined dataset* and its associated *Metadata*.

As described in Sect. 3.3, the *Metadata* of the Refined dataset has three different components:

- *Source metadata*, as described above, to describe the Source data.
- *Execution metadata*: following the PROV model described in Sect. 5.2, the pipeline execution generates metadata related to the *Execute* and *Evaluate* actions, which associate metadata about the Configuration of the pipeline for the preparation and about the metrics related to the actual time and cost of the execution.
- *Provenance Metadata*: the metadata generated in the data preparation pipeline design phase are also associated as metadata. They provide two kinds of information: first, information about the preparation pipeline generation, including design choices; second, an estimation of the quality of the resulting dataset, derived from the assessment performed on the sample dataset used during the pipeline design.

Finally, we note that the pipeline configuration might need to be periodically reassessed by the data analyst. This is particularly necessary if the characteristics of the source data vary in time or if constraints are violated. This reassessment can result in the need to further revise the configuration, starting from the Pipeline revision step in the Data Preparation Pipeline Design. The data analyst has different possible countermeasures:

- revise the initial goals;
- revise the constraints;
- change the data source or the number of items of the Sample Annotated Dataset;
- provide a different pipeline configuration.

Once the countermeasures are applied, the methodology is reiterated.

7 IMPLEMENTATION OF THE METHODOLOGY

In the current implementation of the methodology, the goal of the pipeline is to extract images from social media, in particular Twitter posts, in case of emergency events. In previous work, some case studies were developed and the pipeline configuration was demonstrated to be a critical task [8, 23]. The risk connected to using automatic classification tools configured with high confidence levels is to discard potentially useful images, so an adequate level of confidence must be evaluated. In addition, in different cases, different image classifiers must be used, depending on the type of emergency being analyzed. So a second issue in this domain is the selection of the appropriate components depending on the case being investigated.

In Section 7.1 we illustrate the infrastructure to support the pipeline configuration with the proposed methodology and in Section 7.2 the tools developed for the validation phase.

In the current preliminary implementation of the pipeline design methodology, we leverage on the previous work for manual pipeline configuration and we implemented specific design tools for the validation.

7.1 Pipeline configuration environment

The approach developed in the Crowd4SDG project[2] for pipeline configuration offers a configuration tool, named VisuaCit after Visual Citizen, to data analysts to select the search keywords, the filtering components, and their configuration parameters. The VisualCit tool, described in [7], together with other components from the Citizen Science Solution Kit (CSSK) of Crowd4SDG, provides an interactive interface for the following actions: selecting search keywords, selecting components from a library of components and associating a confidence level to them, geolocating image from the textual post contents. After a satisfactory configuration has been identified, a configuration file is created and the services used in the interactive interface can be invoked as REST web services through the VisualCit APIs, using the selected configuration. Other tools from the CSSK allow the creation of interfaces for annotating posts and classifying them with the help of citizen scientists.

We illustrate here briefly the main functionalities used during pipeline configuration. For selecting the search keywords, VisualCit[3] allows entering search keyword (OR/ANDed) and suggests new possible keywords based on selected interesting posts by the pipeline configurator. In this way, a sample can be extracted from the selected social media source.

Components to be applied in the pipeline in the data preparation phase can be selected from a catalog of data reduction components, which include selecting posts according to image classification, eliminating similar images, and eliminating not-safe-for-work images that have to be analyzed by citizen scientists. Data augmentation components include geolocating posts or user locations and geocoding them for posts that are not already natively geolocated.

During the pipeline configuration phase, the sample dataset is enriched with annotations that are used in the pipeline validation phase. The annotations are performed by manual annotators using the Citizen Science Project Builder[4] component of the Crowd4SDG CSSK [7]. The project builder allows the easy creation of an interactive dataset annotation environment. The estimated time for setting up such an environment using the Project Builder, including the definition of tasks for the crowd and loading the sample dataset from VisualCit is around 30 minutes. For precision and recall evaluations, annotations are needed from at least three different annotators. For a single evaluation task, the average time is 54 minutes for each annotator for a sample of 1000 images.

7.2 Pipeline validation environment

In the pipeline validation phase, the configured pipeline is executed on the selected sample. At the end of this process, the provenance metadata for the configuration is created, including the pipeline configuration, the chosen confidence levels, and overall values for precision, recall, reduction rate, and execution times. These metadata must satisfy the defined constraints for the pipeline configuration to be considered acceptable.

During the assessment phase, some analysis tools are provided to evaluate i) whether each component contributes effectively to the data preparation; ii) the order of execution of the components; iii) the component configuration parameters (in particular confidence levels for classifiers); iv) if sampling filters have to be introduced to comply with the project constraints in terms of execution times. For assessing the execution times, the mean and standard deviation times per item are estimated for each component. For data quality, the data analyst is provided with graphs indicating precision, recall, and a reduced rate for each component with different confidence levels. Additionally, an overall assessment of the precision and recall, and of execution time, for the execution of the pipeline in a given configuration is provided. These graphs help the data

[2]http://crowd4sdg.eu
[3]http://visualcit.polimi.it:20003/
[4]https://lab.citizenscience.ch/

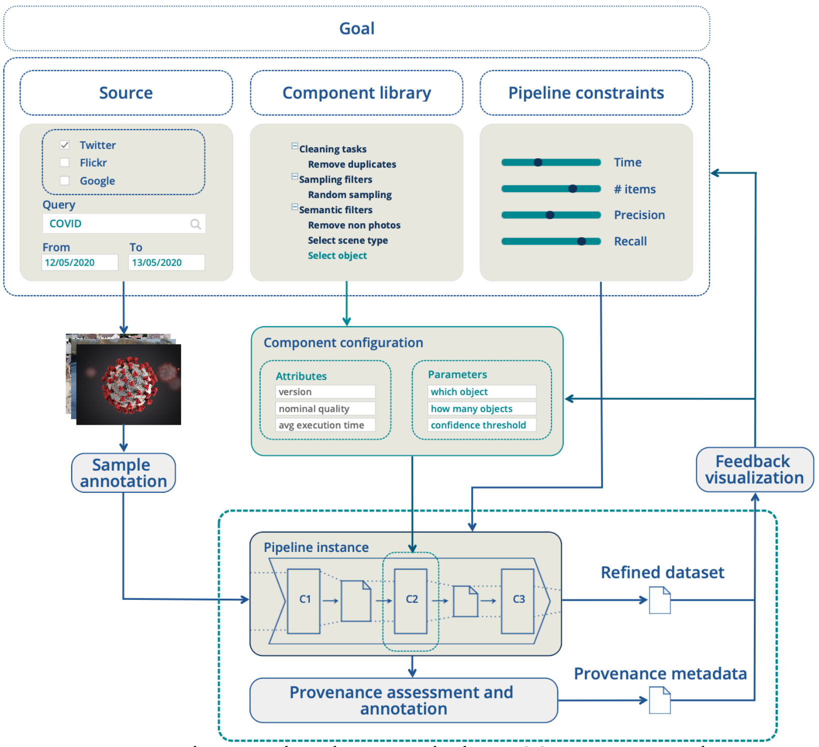

Fig. 4. Data analysis pipeline design applied to a COVID-19 image dataset

analyst to assess if the configuration is sub-optimal and provide clues for its revision. An interface for providing these functionalities has been developed in Python using Colab and it is openly accessible[5]. This prototype loads annotated data, executes a custom pipeline configuration, and computes the aforementioned metrics, displaying them together with constraints.

The pipeline configuration assessment is illustrated in detail in a case study in Section 8, discussing how configuration choices can be assessed.

8 CASE STUDY

The case study presented in this section is based on the analysis of social distancing indicators during the COVID-19 pandemic, borrowing from the experience gained in [23]. The purpose is to generate a dataset with relevant images that the data analyst can use directly or submit to crowd workers, in order to analyze if social distancing guidelines are respected in a specific context. The goal given by the data analyst for the data preparation phase is to *"select photos with at least two persons in a public space"*. The Data Preparation Pipeline is used to obtain a Refined Dataset to avoid the submission of useless tasks to crowd workers who have to evaluate social distance behaviors. The overall design approach is shown in Fig. 4.

8.1 Pipeline configuration

The specified goal drives the three steps composing the pipeline configuration.

8.1.1 Data collection and annotation. In this step, the data analyst needs to provide the source description. As described in Definition 4.1, this consists of three parts whose details are stored in

[5]http://tiny.cc/d213vz

ID	*Component*	*Description*
C_1	Photo	Photo filter aims to isolate real-world photos from other images, using a fine-tuned VGG19 model pre-trained on ImageNet [12].
C_2	TwoPersons	Detection of persons is achieved using YOLOv5[6]. The component is configured to look for images with at least two persons.
C_3	PublicPlace	Public place filtering uses a ResNet architecture trained on Places365 [33], with a threshold on probability for a set of public places.

Table 1. Description of the components selected in the pipeline configuration phase

provenance metadata: (i) the social media from which the dataset has to be crawled, in this case, Twitter; (ii) the query consisting of the set of keywords used to select the items, in this case, related to COVID'19 (e.g., *{coronavirus, corona, virus, covid, covid19, covid-19, flu, Wuhan, Coronaviridae, N95}*); (iii) the time interval, including start and end times, here from August 17, 2020 to August 23, 2020 included.

```
...
agent(pipeline:Source, [prov:type="pipeline:Source", prov:organization="Twitter"])
entity(pipeline:crawler, [prov:type="pipeline:Crawler", pipeline:engine="Twitter-API",
pipeline:version="v2"])
activity(pipeline:crawling1, 2020-08-17T00:00:00, 2020-08-24T00:00:00,
[prov:type="pipeline:Crawling",
pipeline:query="coronavirus"])
wasAssociatedWith(pipeline:crawling1, pipeline:data_analyst, -)
used(pipeline:crawling1, pipeline:crawler, -)
...
```

The source outputs approximately 500,000 images per week. According to the proposed methodology, a sample dataset consisting of 918 images is extracted. In order to obtain a well-grounded truth for annotations, the images composing the sample dataset were each reviewed by three crowd workers, annotating the item as relevant if it is a photo with at least two persons in a public space, as described in the goal. To account for disagreements between crowd workers, an item was marked as relevant when the agreement among annotators was above or equal to 66%. The provenance metadata stored the information related to the post belonging to the sample dataset (media url, post id, etc..), to the annotators, and the result of the annotations.

```
...
entity(pipeline:post1, [prov:type="pipeline:Post",pipeline:post_id="1296474485688852480",
prov:atTime="2020-08-20 17:49:39",
pipeline:media_url="https://pbs.twimg.com/media/Ef4ArKdXkAIbZCV.jpg"])
wasDerivedFrom(pipeline:sample_dataset, pipeline:post1, -, -, -)
wasGeneratedBy(pipeline:sample_dataset, pipeline:crawling1, -)
agent(pipeline:annotator1, [prov:type="pipeline:Annotator"])
wasAssociatedWith(pipeline:annotate, pipeline:annotator1, -)
entity(pipeline:annotated_dataset, [prov:type="pipeline:AnnotatedDataset",
pipeline:annotations=('Valid')])
wasGeneratedBy(pipeline:annotated_dataset, pipeline:annotate, -)
...
```

8.1.2 Components selection and workflow definition. The data analyst selects the components of the pipeline from the Components Library. In the considered context, the target function can be decomposed as the logical conjunction of three separate sub-goals: "is a photo", "contains two persons", "displays a public space". These components leverage established deep learning architectures and are briefly described in Tab. 1. For each component, the data analyst specifies the parameters. Initially, a confidence threshold of 0.7 is specified for all components. Additionally, the order of execution is set as [C_1: *Photo*, C_2: *PublicPlace*, C_3: *TwoPersons*]. Provenance metadata store the components of the pipeline, their order, and settings.

```
...
entity(pipeline:component_Photo, [prov:type="pipeline:Component",
pipeline:version="VGG19-ImageNet"])
entity(pipeline:configurationInitial, [prov:type="pipeline:Configuration",
pipeline:component_order=('component_Photo', 'component_PublicPlace', 'component_TwoPerson'),
pipeline:parameters=('threshold', 0.7, 0.7, 0.7)])
wasAttributedTo(pipeline:configurationInitial, pipeline:data_analyst)
activity(pipeline:generate_configuration1, -, -, [prov:type="pipeline:GenerateConfiguration"])
used(pipeline:execute_configuration1, pipeline:component_Photo, -)
used(pipeline:generate_configuration1, pipeline:component_Photo, -)
...
```

8.1.3 Constraints definition. Here we assume that our aim is to have as a constraint for the refined dataset 50,000 twitter images to be analyzed by the crowd each week, i.e., a reduction rate of 90% w.r.t. the source data. An additional requirement is to have a precision of at least 75% in the refined dataset. Both settings are stored in provenance metadata.

```
...
entity(pipeline:constraint_Size,[prov:type="pipeline:Constraint",pipeline:desidered_size=50000])
wasAttributedTo(pipeline:constraint_Size, pipeline:data_analyst)
used(pipeline:generate_configuration1, pipeline:constraint_Size, -)
...
```

8.2 Pipeline validation

8.2.1 Pipeline execution and assessment. Once a preliminary pipeline has been defined, it is executed on the Sample Annotated Dataset. To enable the execution of the pipeline, the components are exposed through a web service. The web service receives as input the name of the component to be invoked and the set of the URLs of the images to be analyzed and it returns, for each item, the annotation provided by the component (relevant/not relevant) and its confidence.

The output dataset originated by the pipeline execution is then used to assess quality, cost, and time metrics, by comparing the annotations obtained with the ones in the Sample Annotated Dataset. The satisfaction of the constraints can then be verified. During this phase, the information on the execution of the components, together with their parameters, metrics and the results of the pipeline are tracked.

8.2.2 Pipeline revision. As an output for the data analyst, the *Feedback dashboard* (see Sect. 5.3) provides several statistics that are computed when the components are applied to the Sample Annotated Dataset. In Tab. 2 we report summary statistics for the processing time of the components on the Sample Annotated Dataset[7].

Insights on the behavior of the three components with different confidence thresholds are derived using annotated data and provenance metadata. A visual intuition about how the parameters (e.g., confidence thresholds) could be tuned in order to match the pipeline constraints is provided to the data analyst and is shown in Fig. 5. This information is essential for the data analyst, since the precision/recall trade-offs will usually take different shapes in real-world scenarios, depending on the dataset. The data analyst can exploit this information to revise the confidence threshold of the single components.

The *Feedback dashboard* is aimed at understanding the behaviour of the pipeline when varying its parameters. As mentioned before, a grid exploration of the variable space is performed in order to estimate the Pareto frontier for the objectives. This functionality can be appreciated in Fig. 6,

[7]These measures also include processing overheads, such as network communication and request parsing.

Measure	*Photo*	*TwoPersons*	*PublicPlace*
Mean	45 ms	24 ms	37 ms
StdDev	2 ms	3 ms	2 ms

Table 2. Component execution time statistics, estimated on 10 runs, for one item

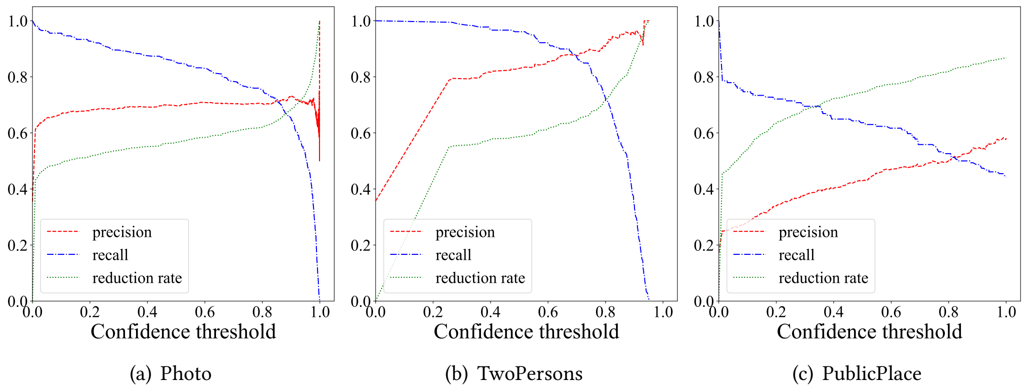

Fig. 5. Precision, recall and reduction rate responses to confidence threshold

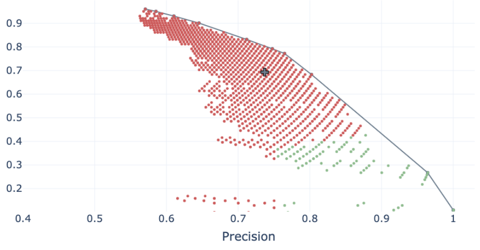

Fig. 6. Confidence grid search for precision, recall and reduction rate. The cross marker represents the initial configuration (0.7 confidence threshold). Solutions that satisfy the constraints in a lighter shade.

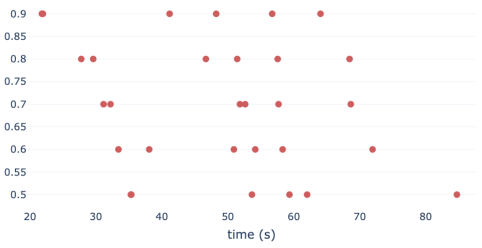

Fig. 7. Execution times for the same pipeline while varying components ordering. Horizontal lines represent different confidence thresholds.

representing the pipeline performance in terms of precision and recall when changing the pipeline parameters. If applicable, admissible regions for the required reduction rate are highlighted. With this tool, moving from sub-optimal solutions to better ones is just a matter of moving a marker in the space of pipeline configurations towards the desired region of the Pareto frontier, subject to the constraints.

In our example, the resulting reduction rate is approximately 81%, therefore the number of output items violates the constraints according to the pipeline assessment. The data analyst could evaluate improvements that increment both reduction rate and precision: for example, there is room for enhancement rising the confidence threshold for the *TwoPersons* and *PublicPlaces* components, while the same operation for the Photo component appears less convenient.

Finally, in light of reduction rates and component execution times, we can possibly suggest a different ordering, following the intuition that faster and more selective components should be placed upstream in the pipeline. With such heuristics, alternative orderings can be selected by the data analyst, revising the initial pipeline configuration. Executing this new pipeline on the sample dataset, we demonstrated a significant execution time decrease (down to 33% of the original execution time in our experiments) without affecting the precision and recall values. Fig. 7

empirically shows the effect of changing the ordering of the components. Each row corresponds to a single confidence threshold, that is fixed for all the components in the pipeline. The different points along each row represent the execution times with different orderings of the components. Consistently across different orderings, in our experimental setup, the best configuration was [C_2: *TwoPersons*, C_1: *Photo*, C_3: *PublicPlace*], and the worst was [C_3: *PublicPlace*, C_1: *Photo*, C_2: *TwoPersons*].

An extract of the provenance model obtained by executing the pipeline on a small dataset can be retrieved and visualized from the shared repository[8]. The repository also contains the sample dataset, aggregated annotations, and the dataset in output of each of the components of the pipeline, thus enabling the replication of the results.

9 RELATED WORK

Data ecosystems are emerging to indicate infrastructures in which several actors can share data and collaborate with the goal of creating value for all participants. New requirements are emerging towards enhancing trust in data, including transparency of the data generation process [15], and it is important to associate quality properties to data and data processing for enhancing the (re)usability of datasets. When considering datasets derived from social media, the number of relevant posts for a specific investigation could be limited, e.g., when natural disasters strike and social media are used for getting a rapid awareness on the event [28] in which first-hand evidence is limited, and most of the posts just refer to the event in a generic way. Another factor is the presence of fake news or bot-generated posts, which may have heavy implications on the results of analyses of news on social media [18, 24]. As a result, datasets in general, and specifically those derived from social media as data sources, need to be preprocessed before using them, to derive a dataset that is fit for use for the analyses at hand. The data preparation activities can be performed both automatically and manually. An overview of data science processes is provided in [5], where the main steps of the data lifecycle are presented, and the importance of data curation, tracking, caring, and metadata are discussed. On the other hand, the attention to the data preparation process in such contexts and their implication on the quality of the results are gaining more importance [22]. In [2] different combinations of machine learning classifiers and crowdsourcing for data annotations are discussed, with an empirical analysis on the suggested combinations depending on the analysis requirements. Several tools (e.g., RapidMiner[9], Knime[10]) offer a platform for supporting also non-expert users in the design of preprocessing pipelines. They allow users to select data preparation components and build a workflow through a visual, drag and drop style environment without the need to have programming skills since coding is not required. In this way, it is possible to achieve fast prototyping and short feedback cycles. However, such tools cannot be used in our context since they are able to deal with structured and semistructured datasets and not with images. Furthermore, they do not offer a proactive support for the selection of a better configuration: they just offer an environment in which the user can try different configurations and visualize the related results. In emergency situations, when awareness data are derived from social media, the data preparation phase assumes a particularly important aspect. In fact, the total workload of emergency operators is constrained and some studies analyze the correlation between workload and recall of highest-ranked alerts [26]. The social media data are usually very noisy, with a limited number of relevant information, so automatic filtering must be used for reducing the number of irrelevant posts [3]. In [1], the authors focus on characterizing machine learning processes, proposing a maturity model and emphasizing

[8]https://bit.ly/30CNVcU
[9]https://rapidminer.com/
[10]https://www.knime.com/

the need of assessing the accuracy of the results of a pipeline. An initial classification of issues related to combining automatic and manual data collection and analysis activities in the framework of Citizen Science initiatives is proposed in [13]. Further analysis and research work is needed to develop a systematic approach to evaluate the different alternatives in deriving datasets for analysis tasks. The fact that different data preparation pipelines may lead to different analysis results has been also highlighted in [6], where the author proposes the Learn2Clean method to identify the optimal sequence of tasks for the maximization of the quality of the results. In this approach, the feasibility of the pipeline in terms of cost and execution time has not been considered.

Pipelines for social good projects are analyzed in [30]. In particular, the paper discusses the problems of data collection and integration and of managing a large number of annotators in crisis events. In [17], in the E2mC project, the use social media posts for emergencies has been studied, focusing on extracting images for awareness purposes. In such an application, the need to allocate crowdsourcing efforts and to reduce the number of non-relevant posts arises. As shown in some examples of this paper, in [23] we presented a pipeline in which the preparation phases are augmented with some automatic ML-based filters to remove non-relevant images in a generic way. In the paper, we did not focus on the process of systematically designing the pipeline configuration, focusing only on functionality.

Being able to replicate results and comprehensively understand their meaning is also an increasingly central issue [20]. In every complex data processing chain, it is important to follow the FAIR (Findability, Accessibility, Interoperability, and Reusability) principles [32] to keep track of both data and experiment procedures [10, 20]. Data provenance can address these requirements while ensuring high-quality analysis, as in the case of explainable Artificial Intelligence (AI) [29]. It can also be used actively to infer valuable truths for optimizing the applications, as in the case of Machine Learning (ML) tasks or scientific pipelines [10, 25]. The W3C PROV data model [4] and the data sheets [14] are the primary directives to define the structure of the provenance metadata. The W3C PROV data model defines a collection of concepts and relationships between them that are responsible for the generation of a piece of data from a business process perspective. Instead, in the case of datasheets, indications are provided to formulate questions about the data generation and answers are collected in a structured text file. In this work, we decided to adopt the W3C PROV data model instead of the datasheets to keep track of provenance since the latter is a poorly machine-readable format and would still require an effort to determine their structure.

We recently proposed an initial discussion on requirements to model data analysis pipelines [9], which has been fully developed into a methodology and support tools in the current paper. In this paper, we further develop the approach focusing on supporting design choices for designing a data preparation system able to associate a formal annotation of their properties and derivation to resulting datasets. We also propose an approach to automatically create data provenance metadata based on the characteristics of the designed pipelines and the intrinsic characteristics of the datasets being preprocessed. These metadata are the basis for providing suggestions to data analysis pipeline designers.

10 CONCLUDING REMARKS AND FUTURE WORK

In this paper, we have proposed a systematic approach for designing data preparation pipelines. The approach involves a human-in-the-loop solution, based on the definition of a process for configuring and validating the data preparation pipeline on a sample dataset extracted from the target dataset. The data analyst is actively involved in all the phases of the process, from defining the objective of the analysis to selecting and configuring the components of the pipeline and validating the results. Automatic support is provided to the user in all these stages through a series of metadata and annotations that can be used to evaluate and/or estimate the effectiveness of the pipeline for the specific objective, as well as suggest alternative and more effective configurations. To this end, a series of metrics are defined to evaluate the effectiveness of the pipeline and its components in both a general and context-dependent manner, based on time, cost, and quality assessment.

Furthermore, the proposed approach promotes the trustworthiness and transparency of the dataset generation by integrating the dataset with metadata, proposing a provenance data model to trace the source of the input dataset and all the transformations performed on it to obtain the target dataset. We believe that provenance metadata are essential to enable the reusability of both the refined datasets generated and the pipeline in similar contexts.

In the future, we will refine the component library with additional component types (in this paper we have mainly focused on data reduction components) and define specific quality metrics for each of them. We will also introduce a self-adaptive approach, capable of providing more refined and complex pipeline configuration suggestions, using a goal-oriented approach, and supporting some of the design choices, while maintaining a human-in-the-loop approach. Starting with abstract goals, we aim to automatically suggest an appropriate set of components to the data analyst and automatically optimize the workflow and parameters, taking into account the quality and performance constraints expressed. To this end, we will use the provenance metadata to obtain useful information from previous configurations. Finally, we aim to improve the data crawling phase, enriching the keywords suggested by the data analyst and removing biases from the generated datasets in order to increase the fairness of the decision-making process.

ACKNOWLEDGMENTS

This work was funded by the European Commission H2020 project Crowd4SDG "Citizen Science for Monitoring Climate Impacts and Achieving Climate Resilience", #872944. This work expresses the opinions of the authors and not necessarily those of the European Commission. The European Commission is not liable for any use that may be made of the information contained in this work.

REFERENCES

[1] Rama Akkiraju et al. 2020. Characterizing Machine Learning Processes: A Maturity Framework. In *Proc. BPM Conf. (LNCS, Vol. 12168)*. Springer, 17–31.

[2] Samreen Anjum, Ambika Verma, Brandon Dang, and Danna Gurari. 2021. Exploring the Use of Deep Learning with Crowdsourcing to Annotate Images. *Human Computation* 8, 2 (2021), 76–106.

[3] Sara Barozzi, Jose Luis Fernandez-Marquez, Amudha Ravi Shankar, and Barbara Pernici. 2019. Filtering images extracted from social media in the response phase of emergency events. In *Proc. ISCRAM*. 768–779.

[4] Khalid Belhajjame, Reza B'Far, James Cheney, Sam Coppens, Stephen Cresswell, Yolanda Gil, Paul Groth, Graham Klyne, Timothy Lebo, Jim McCusker, et al. 2013. PROV-DM: The PROV data model. *W3C Recomm.* 14 (2013), 15–16.

[5] Francine Berman, Rob Rutenbar, Brent Hailpern, Henrik Christensen, Susan Davidson, Deborah Estrin, Michael Franklin, Margaret Martonosi, Padma Raghavan, and Victoria Stodden. 2018. Realizing the potential of data science. *Commun. ACM* 61, 4 (2018), 67–72.

[6] Laure Berti-Équille. 2019. Learn2Clean: Optimizing the Sequence of Tasks for Web Data Preparation. In *Proc. WWW Conf.* ACM, 2580–2586.

[7] Carlo Bono, Mehmet Oğuz Mülâyim, Cinzia Cappiello, Mark Carman, Jesus Cerquides, Jose Luis Fernandez-Marquez, Rosy Mondardini, Edoardo Ramalli, and Barbara Pernici. 2022. Analyzing social media with crowdsourcing in Crowd4SDG. https://doi.org/10.48550/ARXIV.2208.02689

[8] Carlo Bono, Barbara Pernici, Jose Luis Fernandez-Marquez, Amudha Ravi Shankar, Mehmet Oğuz Mülâyim, and Edoardo Nemni. 2022. TriggerCit: Early Flood Alerting using Twitter and Geolocation–a comparison with alternative sources. In *ISCRAM 2022 Conference Proceedings – 19th International Conference on Information Systems for Crisis Response and Management*, Rob Grace and Hossein Baharmand (Eds.). Tarbes, France, 674–686.

[9] Cinzia Cappiello, Barbara Pernici, and Monica Vitali. 2021. Modeling Adaptive Data Analysis Pipelines for Crowd-Enhanced Processes. In *Conceptual Modeling - 40th Intl. Conf., ER 2021, Virtual Event, Proceedings (Lecture Notes in Computer Science, Vol. 13011)*, Aditya K. Ghose, Jennifer Horkoff, Vítor E. Silva Souza, Jeffrey Parsons, and Joerg Evermann (Eds.). Springer, 25–35.

[10] Adriane Chapman, Paolo Missier, Giulia Simonelli, and Riccardo Torlone. 2020. Capturing and querying fine-grained provenance of preprocessing pipelines in data science. *Proceedings of the VLDB Endowment* 14, 4 (2020), 507–520.

[11] Kyle Cranmer, Lukas Heinrich, Roger Jones, David M South, ATLAS collaboration, et al. 2015. Analysis preservation in ATLAS. In *Journal of Physics: Conference Series*, Vol. 664. IOP Publishing, 1–5.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *IEEE Conf. on Computer Vision and Pattern Recognition*. IEEE, 248–255.

[13] Chiara Franzoni, Marion Poetz, and Henry Sauermann. 2022. Crowds, citizens, and science: a multi-dimensional framework and agenda for future research. *Industry and Innovation* 29, 2 (2022), 251–284.

[14] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. 2021. Datasheets for datasets. *Commun. ACM* 64, 12 (2021), 86–92.

[15] Sandra Geisler, Maria-Esther Vidal, Cinzia Cappiello, Bernadette Farias Lóscio, Avigdor Gal, Matthias Jarke, Maurizio Lenzerini, Paolo Missier, Boris Otto, Elda Paja, Barbara Pernici, and Jakob Rehof. 2022. Knowledge-driven Data Ecosystems Towards Data Transparency. *ACM Journal of Data and Information Quality* 14, 1:3 (March 2022), 1–12.

[16] Alex Gorelik. 2019. *The Enteprise Big Data Lake.* O' Reilly.

[17] Clemens Havas et al. 2017. E2mC: Improving Emergency Management Service Practice through Social Media and Crowdsourcing Analysis in Near Real Time. *Sensors* 17, 12 (2017), 2766.

[18] Maryam Heidari, James H Jones, and Ozlem Uzuner. 2020. Deep Contextualized Word Embedding for Text-based Online User Profiling to Detect Social Bots on Twitter. In *2020 Intl. Conf. on Data Mining Workshops (ICDMW)*. 480–487.

[19] Gary T Henry. 1990. *Practical sampling*. Vol. 21. Sage.

[20] Melanie Herschel, Ralf Diestelkämper, and Houssem Ben Lahmar. 2017. A survey on provenance: What for? What form? What from? *The VLDB Journal* 26, 6 (2017), 881–906.

[21] Deniz Iren and Semih Bilgen. 2014. Cost of quality in crowdsourcing. *Human Computation* 1, 2 (2014), 283–314.

[22] Felix Naumann. 2021. Bad Files, Bad Data, Bad Results: Data Quality and Data Preparation. In *Proc. CAiSE*. xxii–xxiv.

[23] Virginia Negri, Dario Scuratti, Stefano Agresti, Donya Rooein, Gabriele Scalia, Amudha Ravi Shankar, Jose Luis Fernandez Marquez, Mark James Carman, and Barbara Pernici. 2021. Image-based social sensing: combining AI and the crowd to mine policy-adherence indicators from Twitter. In *IEEE/ACM 43rd Intl. Conf. on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, 92–101.

[24] Francesco Pierri and Stefano Ceri. 2019. False news on social media: a data-driven survey. *ACM Sigmod Record* 48, 2 (2019), 18–27.

[25] Débora Pina, Liliane Kunstmann, Daniel de Oliveira, Patrick Valduriez, and Marta Mattoso. 2020. Provenance Supporting Hyperparameter Analysis in Deep Neural Networks. In *Provenance and Annotation of Data and Processes*. Springer, 20–38.

[26] Hemant Purohit, Carlos Castillo, Muhammad Imran, and Rahul Pandey. 2018. Ranking of social media alerts with workload bounds in emergency operation centers. In *Proc. Conf. on Web Intelligence (WI)*. IEEE, 206–213.

[27] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2012. Tweet analysis for real-time event detection and earthquake reporting system development. *IEEE Transactions on Knowledge and Data Engineering* 25, 4 (2012), 919–931.

[28] Gabriele Scalia, Chiara Francalanci, and Barbara Pernici. 2022. CIME: Context-aware geolocation of emergency-related posts. *GeoInformatica* 26 (2022), 125–157.

[29] Stefanie Scherzinger, Christin Seifert, and Lena Wiese. 2019. The Best of both Worlds: Challenges in Linking Provenance and Explainability in Distributed Machine Learning. In *IEEE 39th Intl. Conf. on Distr. Computing Systems*. 1620–1629.

[30] Christoph Scheunemann, Julian Naumann, Max Eichler, Kevin Stowe, and Iryna Gurevych. 2020. Data Collection and Annotation Pipeline for Social Good Projects. In *Proc. of the AAAI Fall 2020 AI for Social Good Symposium*. 1–7.

[31] Victoria Stodden. 2020. The data science life cycle: a disciplined approach to advancing data science as a science. *Commun. ACM* 63, 7 (2020), 58–66.

[32] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data* 3, 1 (2016), 1–9.

[33] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. 2018. Places: A 10 Million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 6 (2018), 1452–1464.