This project has received funding from the European Union's Horizon 2020 research and innovation programme under **grant agreement No 780355**

European Commission

FANDANGO

D2.4 TECHNICAL REQUIREMENTS (PLATFORM AND SERVICES REQUIREMENTS)

Deliverable No.:	D2.4
Deliverable Title:	Technical Requirements (Platforms and services requirements)
Project Acronym:	Fandango
Project Full Title:	FAke News discovery and propagation from big Data and artificial intelliGence Operations
Grant Agreement No.:	780355
Work Package No.:	WP2
Work Package Name:	Data Access, Interoperability and user requirements
Responsible Author(s):	Daniele Vannella, Camila Garcia, Fulvio D'Antonio
Date:	30.06.2019
Status:	v1.0
Deliverable type:	REPORT
Distribution:	PUBLIC

Revision History

Version	Date	Modified by	Comments
V0.1	20.05.2019	Camila Garcia(LvT)	First draft, contribution 1.1
V0.2	20.05.2019	Daniele Vannella(LvT) Fulvio D'Antonio(LvT)	Internal review
V0.3	20.06.2019	Panagiotis Stalidis(CERTH)	contributions, 2 and 3
V0.4	15.06.2019	David Martìn (UPM)	contributions, 1.2
V0.5	27.06.2019	Jeferson Zanim (SIREN)	contributions, 2 and 3
V0.6	27.06.2019	Monica Franceschini (ENG)	contributions
V0.7	28.06.2019	Daniele Vannella	Quality check
V1.0	30.06.2019	Fulvio D'Antonio	Final review

TABLE OF CONTENTS

ABBREVIATIONS

ABBREVIATION	DESCRIPTION
H2020	Horizon 2020
EC	European Commission
WP	Work Package
EU	European Union

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

This deliverable is an official document of the FANDANGO project funded by the European Union's Horizon 2020 (H2020) research and innovation programme under grant agreement No 780355. It is a public report that describes the technical requirements for FANDANGO platform and services.

This document reports the technologies needed to support the user requirements specified in D2.3, in which users has defined their needs to fulfil the final goal of the project.

The main objective of this deliverable is to provide evidence about all aspects in terms of software and information technologies. The user requirements were gathered from professional journalists and analogue figures and reflect their specific needs.

A narrative use case was added to provide an example of the requirements that are gathered in D2.3 user requirements and in the pilot execution and evaluating plans.

1. INTRODUCTION

FANDANGO's final goal is to integrate an information verification system, able to analyze different typologies of data, correlated with a news like images, video, text and metadata(source, author, etc). To achieve such a goal, several different tasks and functionalities need to be developed and integrated to provide a final tool where users can navigate and obtain useful results.

In particular, it has to be underlined how meeting users requirements align with project tasks and how fandango's platform and system will solve all the user's needs at a high level. Morevore, there has to be another side of requirements analysis dealing with functional requirements, which define the desired output or behaviour of a system

The template for the following paragraphs will consider key tasks to describe main processes, in terms of software development and systems engineering. Those key processes represent a subset of functionalities to solve user's needs described in D2.3 and in the document called pilot execution and evaluating test.

USE CASES IN USER REQUIREMENTS	KEY TASKS INVOLVED
Verify an article - This use case describes the process a journalist should be following when verifying whether an article is trustworthy or rather considered as disinformation.	**News trustworthiness with ML**
Verify a video - This use case describes the process a journalist should be following when verifying whether a video is trustworthy or rather considered as disinformation. **Verify an image** - This use case describes the process a journalist should be following when verifying whether an image is trustworthy or rather considered as disinformation.	**Image and video verification**
Verify a claim - This use case describes textual matching in the process a journalist should be following when verifying whether a claim is trustworthy or rather considered as disinformation.	**Claims analysis**

Table 1 - user requirements connected with key tasks in Fandango

The template to follow in order to describe functionalities/key tasks is:

Goal

The final result the process provide or is connected to.

Pre-condition(user requirement addressed)

User requirement pre-condition or action/s that triggers the process.

Chosen approach

Brief description of the process.

Technologies involved in the process

Softwares, libraries, platform involved in the process.

Non-functional requirements

How the system/module should behave and that it is a constraint upon the systems/modules behaviour. Requirements about resource required, response time, the accuracy of the data, the effort required to make changes in the software(measurement is personnel effort), reliability.

Process Output

Output returns by the process.

PLATFORM OVERVIEW

PLATFORM GENERAL DESCRIPTION

In order to meet all the requested made by end-users in Deliverable 2.3, a new design for fandango's platform architecture has been created. In this new proposal, all the tasks and functionalities have been divided into two main workflows, called online and offline workflow.

The offline workflow is intended as all the processes to ingest data needed to build most of the machine learning, graph models and official data lake for Fandango. In fact the ground truth will be taken from the ingestion process and from the annotation system as well as information to build graphs about authors and organizations/publishers.

The diagram contains all the technologies involved and a full connection among them.

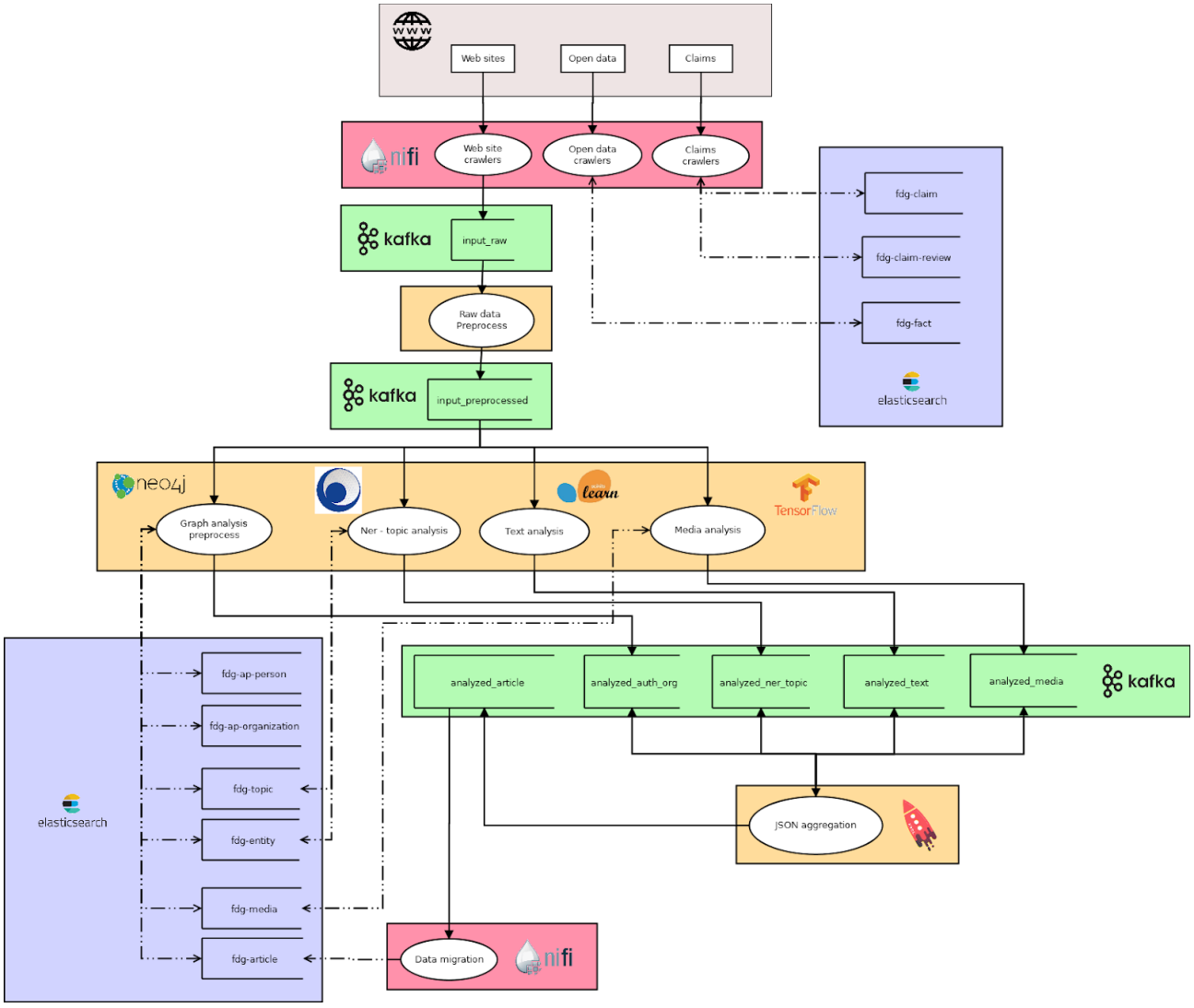

Figure 1 - Offline workflow

On the top of the diagram there are data typologies that are necessary to have all the services integrated in the platform. Kafka is used in order to handle asynchronous returns in the results, since images and videos analysis are much more time consuming than a prediction score calculation in terms of text and graph analysis. At the end, a final document will be ready to be stored on the nosql database ElasticSearch.

The data ingestion is intended as a continuous process, the more information the system can hold and add, the better performances in terms of relations and correlations among articles, authors, publishers, topics and entities will be.

The online flow describes all the services/modules interconnections to be integrated in the web-application.

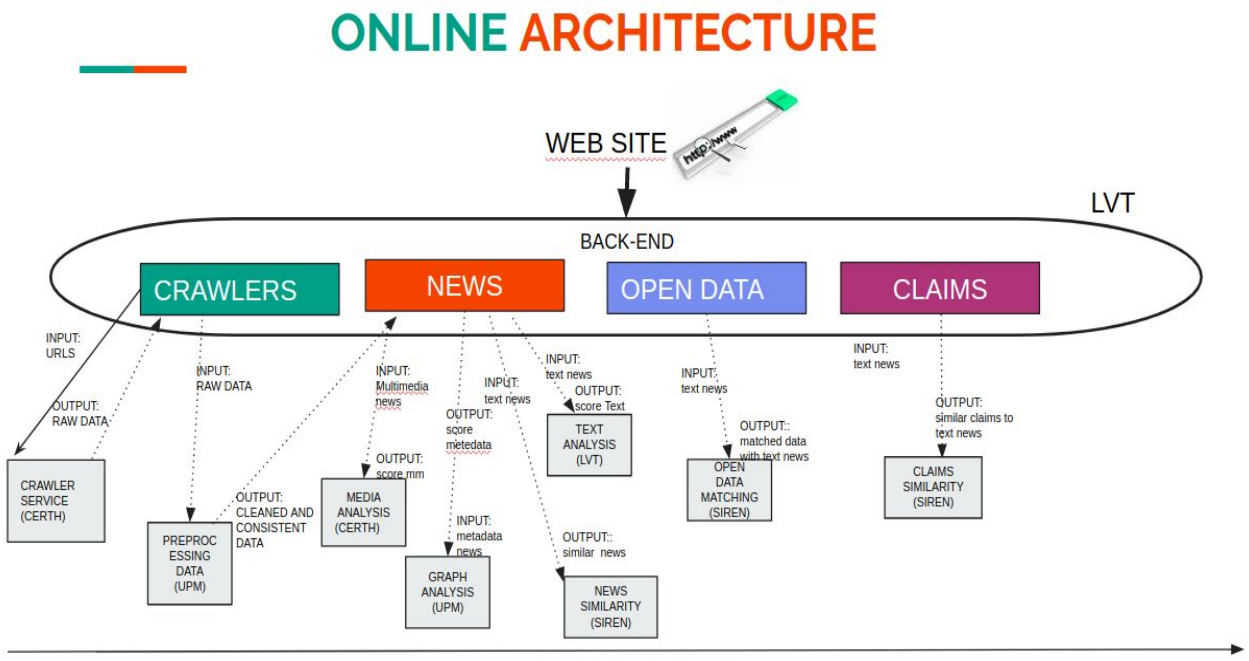

Figure 2 - Online Services Architecture

Through the web-app several processing flows to analyze the trustworthiness of a news it can be initiated: the first of these flows executes sequentially the **Crawler** service, **Pre-Processing**, **Text Analysis**, **Meta Data (Graph) Analysis**, **Open Data Matching** and **Similar News**. The second processing flow retrieves the most similar claims by means of **Entity service** and **Claim Similarity** service. Finally, the last processing flow is executed in order to analyze images or videos by means of the **Media Analysis** service**.**

PLATFORM NON-FUNCTIONAL ASPECTS

The platform set up to support the previously described analytics, both from a batch and an online perspective, enables scalable pipelines and processes. The state of the art of the project is implementation phase, relying on a development environment which benefits from a cloud infrastructure on MS Azure. At this point data is ingested in bulk mode too, in order to collect all of the available raw data to archive historical data for the further analysis. The final amount of raw data to ingest and to implement for the use-cases is still to be defined, as well as the system doesn't have specific SLAs to grant.

Thanks to this configuration the Fandango architecture, which is composed by scalable tools and services, is comfortable to achieve all of the desired performances that the final product must satisfy.

From a technical aspect all of the services are deployed as simple instances that can work in parallel, as well as the rest of the processes. In order to allow many simultaneous users/developers to access the system and set-up/run the services, the platform is delivering the right match of resources in terms of RAM and CPUs. Whenever some fault is detected or more service instances need to be run, exceeding the amount of computational resources, the cluster can be extended both in scale out/scale up mode.

All of the issues that can concern the storage dedicated to memorize raw data as well as the output of the services, plus the disk storage for the final results is fulfilling the current requirements, since the system can benefit from hard disk of the Elasticsearch cluster, local storage within the containers, the persistence layer (relying on disk storage) for the retention in Kafka and, above all for intermediate results Azure Blob Storage, the MS Azure object store, can be used.

Since this is a complex system, composed by several services and tools, many quantitative aspects of the way the whole platform performs, depend on the actual use-cases in terms of processes and storage. Setting a proper prototyping during this phase of the project, can help identifying the perfect hardware provisioning for the final production environment.

1. NEWS TRUSTWORTHINESS WITH ML

Fake news recognition has a particular growing interest as the circulation of misinformation on-line increase. This problem, in Fandango, will be faced in two different ways; on one hand , considering text and title of an article and analysing its syntax and some linguistic features. On the other hand, building a graph to collect and connect authors and organizations in order to apply some graph analysis. In the following two paragraphs both approaches will be described in detail.

1.1. LINGUISTIC APPROACHES

This task focuses on the automatic identification of fake news in online articles, covering 3 contexts (immigration, climate and european context). The linguistic approach deals with different main steps:

- *Training set creation:*

  The training set is the material through which the computer learns how to process information. Machine learning uses algorithms. The idea is that because the machine learning program is so complex and so sophisticated, it uses iterative training on each of those images to eventually be able to recognize features, shapes and even subjects such as people or animals. The training data is absolutely essential to the process – it can be thought of as the “food” the system uses to operate.

- *Feature engineering:*

  Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. If feature engineering is done correctly, it increases the predictive power of machine learning algorithms by creating features from raw data that help facilitate the machine learning process.

- *Model training:*

  Train a model is a process of fit a machine learning algorithm able to infer a result given a set of variable called predictors.

The core activity in this approach is on feature engineering step, based on an exploratory analysis on the identification of linguistic differences in fake and legitimate news content.

The strategy is to build a fake news detection model, extracting a set of linguistic features and considering just features not related directly with words meaning; in particular, each word won’t count as a single feature, it will bring information to the model taking its syntax and stylometric value.

Unlike the most frequent approach in academic field, take always into consideration semantic words connotation in an article , using models like bag of words, term vector or other kind of vectorization

techniques, it shows some issues dealing with cross-domains specification (apply in different contexts like politics, entertainment, finance).

1.1.1. GOAL:

The final aim of this set of processes is to predict whenever an article is reliable or not, analyzing its text and title. In order to achieve it, a machine learning classifier has to be trained and validated with thousand of labeled articles consider as samples. The prediction will be a probability score given a complete new article.

1.1.2. PRECONDITION (USER REQUIREMENTS ADDRESSED):

A journalist needs to analyze the structure of an article and wants to be supported by the machine learning model built to detect news trustworthiness.

State of the art:

To date, there are two important lines of research into the automated classification of genuine and fake. A linguistic approach attempts to identify text properties, such as writing style and content, that can help to discriminate real from fake news articles. The underlying assumption for this approach is that linguistic behaviors such as punctuation usage, word type choices, part-of-speech tags, and emotional valence of a text are rather involuntary and therefore outside of the author's control, thus revealing important insights into the nature of the text.

The best classification performances were achieved with feature sets representing absurdity, punctuation, and grammar.

Other works have used social network activity (e.g., tweets) on a specific news item to assess its credibility, for instance by identifying tweets voicing skepticism about the truthfulness of a claim made in a news article (Hannak et al., 2014; Jin et al., 2014).

Fake news detection is closely related to deception detection (i.e. determining whether or not someone is lying), there are important differences between the two tasks. First, fake news producers usually seek political or financial gain as well as self promotion while deceivers have motivations that are more socially driven such as self protection, conflict or harm avoidance, impression management or identity concealment. Second, they differ significantly in their target and in the form they propagate: fake news items are usually disseminated at larger scale through the Internet and social media whereas deception is more specifically targeted at individuals. However, since both tasks deal with deceptive content, it has been hypothesized that there are linguistic aspects that might be shared between these tasks.

Another important result has been obtained this year, based on the incorporation of hierarchical discourse-level structure of no legitimate and legitimate. This model learns a constructs a discourse-level structure for legitimate/no legitimate news in an automated and data-driven way. This method outperforms baselines like N-grams, LIWC(Linguistic Inquiry and Word Count), BiGRNN-CNN, LSTM of more than 1.5% of accuracy, reaching exactly 82.19.

1.1.3. Chosen Approach:

The first step to consider is the ground truth construction, containing all the samples to train a model. Since all the articles in this set have to attach a label to each sample(legitimate/no legitimate).

In this project, it has been agreed to adopt 3 different methods to annotate document and create the so called training dataset:

Firstly, using existing dataset applied in academic research, this allows to have a comparison with state-of-art solutions. The first two datasets are available from a paper called Exploiting Tri-Relationship for Fake News Detection written by Kasi Shu, Suhang Wang and Huan Liu including online articles whose veracities have been identified by experts. The other two datasets come from kaggle.

Secondly, with a semi-automatic annotation method, consisting in making a list, verified by expert journalists, supporting some sources labeled as legitimate and no-legitimate (fake).

Legitimate Sources	No-legitimate Sources
theguardian.com	ussia-insider.com/en
apnews.com	sputniknews.com
washingtonpost.com	en.news-front.info
wsj.com	express.co.uk/news/politics

Table 1 - Online sources for ground truth

Articles crawled from above sources are labeled with one of the two tags in the columns of the table above. Further, applying a filter, word-to-vect and sense-embedding techniques it's possible to discriminate off topics news.

In particular, the strategy is to retrieve a main meaning for each news, converting an article in a list of vectors, each word is represented by its correlated words, then a centroid can be calculated to generate a principal vector to measure how close the result is to the vectors(filter) representing topics.

Thirdly, manual annotation from users, through an interface linked to an elastic database, 1000 filtered by context articles for each language considered in this project are going to be annotated by expert journalist and using in part as test set.

In order to classify an article as reliable or not, some specific features that can be used for machine learning model training. Those features are taken from a text and a body of an article and can be divided in the following macro-categories:

- Simple frequency features: stopword counter, characters counter, punctuation counter, all of normalized with respect to the length.
- Part of The Speech(POS) features: count adjective, adverbs, conjunctions, verbs, personal nouns.
- Advance frequency features: word average per paragraph, lexical density, mean segment type-token ratio (MSTTR), moving average type-token ratio (MATTR).
- Readability indices.

By using large amounts of labeled articles as a training, extracting and selecting characteristics from that sample, some syntactic features can be used to define a statistical classification model, able to generate a prediction for a new record, that has not been seen by the algorithm yet.

A model completely "context agnostic" means that each word of the text is not considered as predictor variable. Filtering process described above will previously discriminate the macro-context in the training data.

The entire procedure to create the machine learning classifier able to detect fake news, will be repeated using sources and articles in italian, dutch and spanish. In some languages, not all the features of the model can be implemented due to the structure or the complexity of the function and of the language syntax.

1.1.4. TECHNOLOGIES INVOLVED IN THE PROCESS:

Machine learning classifiers will be created in python programming language, using libraries as DS4Biz Predictor, Scikit-learn, Spacy, nltk, polyglot, TreeTagger, tensorflow, keras and PySpark.

1.1.5. PROCESS OUTPUT

It will provide a score that corresponds to the probability of an article to be classified as reliable or unreliable.

An example of the output provided by the analyzer is presented below in a JSON format with 2 keys form:

a) **Identifier**: the unique identifier of the article. This value is provided during the preprocessing step, where the "texthash" is used for this purpose.
b) **TextRating**: this field will be one of the scores calculated, giving a news prediction analysing article body and title.

1.1.6 NON-FUNCTIONAL REQUIREMENTS:

This service has a response time of less than 1 second, so a prediction can be provided almost immediately.

The machine learning classifier will provide a partial fitting, so it's not necessary to implement a retrain phase when new data will be available. Everytime a new batch of articles will be ready to get ingested, the system will automatically add information to the ML model as well as to elasticsearch database.

It has been estimated at least 10000 articles for each language, with a balanced distribution of legitimate and no-legitimate news, in order to create a valid machine learning model to recognize fake news.

1.2. NETWORK APPROACHES

From an end-user perspective, the analysis of text can support them to understand inconsistencies in the manner of writing, the semantic incoherences and related indicators of abnormalities. However, an

exhaustive analysis must also include high-level descriptors. As an example, not only the text is valuable, but also who wrote, where has the article / news been written, and even more deep analysis on metadata information. This ambition is generally tackled by using network approaches, which consists on express the entities in the news environment as a graph of connected nodes that are joint according to the relation among them. In other words, what network approaches do is to implement the data model into a mathematical formulation that allows to perform analysis to extract relevant patterns and hidden relationships to support journalists and end users in the decision making process.

Therefore, the expected output for end users in this context are values "scoring" the degree of reliability or fakeness of an entity. More information on the data model can be found in deliverable D2.2, however, for the sake of simplicity, let's summarize it into 9 type of entities: Author, Publisher, Article, Media_image, Media_Video, topic, Entity, Claim and Claim-reviewers. To clarify the entities topic and Entity refer to the type of objects found in the news content. As an example, a news over US presidential elections has a topics: elections, Donald Trump, Democratic Party, Republican parties...

Fake news recognition has a particular growing interest as the circulation of misinformation on-line increase. This problem would be faced by building a graph to collect and connect authors and organization in order to apply some graph analysis.

1.2.1. Goal:

This task is directly linked to WP4.4 and the main goal is to provide a reliability score for the aforementioned entities based on the information available, the hidden and expressed relationships that a particular entity (i.e. author) might have.

This proposal will be faced from diverse perspectives. Initially, traditional graph processing techniques will be employed. As an example in this group is the successful PageRank algorithm. Furthermore, the research will be focused on applying novel techniques based on Deep Learning on graphs.

1.2.2. Precondition (User requirements Addressed):

The main challenge to successfully apply the mentioned techniques is the need of having updated labelled datasets. For this ambition, the End users (journalists) will support technical teams in the appropriate classification of data sources.

1.2.3. Chosen Approach:

The current approach considers the use of the background created with the support of end users to express the graph as a network of publishers, authors, articles and additional entities according to FANDANGO's data model.

Moreover, a graph is a combination of two main components: nodes (vertices) and edges (links), where in this case, the different elements involved in the data model such as authors publishers or articles are represented as nodes whereas the connections that they have among them are denoted as edges.

Subsequently, different centrality algorithms such as PageRank or Eigenvector Centrality algorithms are used to compute the influence of the different nodes in the network based on their connections.

Furthermore, combining the different scores of the rest of the analyzer, a weighted metric is computed to support the credibility of either an author or a publisher. For instance, whether the articles associated to an author have a high probability of being fake and the corresponding images to such article were detected as fake as well, it is clear that the credibility of such author should be low. Moreover, if such author has a large centrality score according to the PageRank algorithm it means that his/her fakeness influence in the network is relevant and therefore, its final credibility score must be lower.

Finally, a sigmoid function is used to keep the range between 0 and 1 since the pageRank algorithm range is above it and then it is multiplied by 100 to attach the meaning of percentage. A similar procedure is investigated as well for publishers in order to manage the computation of their credibility measure.

1.2.4. Technologies involved in the process:

For the initial analysis, the most suitable, scalable and fast technology for graph signal processing is Neo4J, and therefore is chosen as the core technology to create the graph and perform the personalised queries. This technology is integrated with powerful tools for advance processing tasks. An adaption to python programming language is done for some related tools such as Natural Language Processing, in the processing tasks libraries such as fuzzywuzzy, nltk and numpy will be employed to speed up processing tasks, cleaning information and adapting it to the proposed models.

1.2.5. Process Output:

The user will receive as result a set of scores for the related entities including both the publisher of the article and the set of authors associated to it.

An example of the output provided by the analyzer is presented below in a JSON format. Four keys form the final solution including:

c) **identifier**: the unique identifier of the article. This value is provided during the preprocessing step, where the “texthash” is used for this purpose.
d) **author**: this field contains a list of Elasticsearch identifiers associated to the authors.
e) **publisher**: this field contains a list with the corresponding Elasticsearch unique identifier of the publisher.
f) **authorRating**: this field will provide the credibility score of the authors and it is represented as a list of percentages in the range (0, 100), where the higher the value is, the higher the credibility of such author will be.
g) **publisherRating**: this field contains the final credibility score associated to the publisher of the article. It is presented as a list of values in the range (0, 100) as well indicating the trustworthiness of the publisher.

{"**identifier**": "Media plays a very important role in news articles. Images and videos are often used as corroboration to the story in an article. It is very important for the Fandango project to be able to automatically evaluate media and mark them as authentic or manipulated. Additionally, even if a media file is authentic, the system should also check if the specific media file coincides with the information

existing in the other modalities of the article. In order to make this evaluation, the Fandango system will provide for a number of different descriptors that will help understand the time frame when the media was captured, the location that is depicted and the objects that are displayed.", "**author**": ["MICT-GoBu09LVEn7SCPg"], "**publisher**": ["MYCT-GoBu09LVEn7SiNn"], "**publisherRating**": [10.25], "**authorRating**": [21.45]}

1.2.6. NON-FUNCTIONAL REQUIREMENTS:

This service employes NEO4J as the main graph framework to investigate and determine the scoring of both authors and organizations. Therefore, all the components of the service must be compatible with the updates of NEO4J in order to behave as it is expected. To do so, all the required libraries are stored in a requirements file with the adequate versions that are compatible with NEO4J. However, this procedure is semi-automatic since it is necessary to review the official documentation of NEO4J to ensure the process.

Moreover, in this first approach of the graph analysis, the service uses two main threads to parallelize and thus, to speed up the operations. Hence, the response time of the module is not elevated as it may occurred by doing sequential operations in the graph.

Furthermore, the service is bound by the data quality, in the sense that if there is dirty data in the nodes of the network, the final performance of the graph will not have the expected accuracy.

Modifying the software is straightforward since all the code is grouped in a folder and it is written in pure Python. On the other hand, when generating the docker, the only issues consists of getting the NEO4J version that we preferred and never selects the last one without a previous analysis regarding compatibility.

In terms of reliability, the graph analysis performance uses all the available information of the network so that, at the very beginning the results will be poor due to the lack of information on the graph. However, having a large amount of data in the network will ensure and improve the performance of the service.

2. CLAIMS ANALYSIS

An important criteria for end-users is that Fandango should never make a final decision about the trustworthiness of information. Instead, it should help the journalists in doing that by providing enough information to base their decision on. This is advisable for legal issues, but also because journalists want to have the ultimate responsibility of judging a news article as fake or not. One of the essential elements for that is the verification of Claims inside the article.

One of the challenges identified during the requirement analysis is that to achieve such goals, it is essential to have the required data to investigate content, where claims already checked by various specialized organizations are an important part of the process to assert trustworthiness.

2.1. GOAL:

In order to support the journalists claim analysis process, FANDANGO aim is to provide real-time contextual information about a claim by identifying similar claims that have been already reviewed by trusted sources. Additionally, it will also suggest references to relevant open data sources catalogued in the platform, that could provide factual sources about the topic being investigated.

2.2. PRECONDITION (USER REQUIREMENTS ADDRESSED):

According to Fandango user requirements (deliverable “D2.3 User Requirements”), there are two user stories that are essential for the claim analysis process:

- search for data sources relevant to statements, with FANDANGO displaying official datasets (e.g. from UN/EU/national statistics agencies) that are the most helpful for the journalist doing the verification;
- third-party fact-checks data, where given a statement/claim, Fandango should check it against its internal database of fact-checking reviews, performed by reputable journalistic organisations (e.g. Politifact, Snopes, Pagella Politica), and suggest the most appropriate ones to verify the given claim.

2.3. CHOSEN APPROACH:

To provide the required functionality, FANDANGO will collect data from sources designated by the end-users as appropriate for fact-checking and automatically display relevant content through the web interface based on the current user selection.

Data collection will be done from websites that provide claim reviews, which follows schema.org ClaimReview object structure, through the use of web crawlers, to compose a claim database in FANDANGO’s structure and serve as reference for recommendations.

Open data references will come from a list of relevant sources, curated by the end users and ingested in the platform data repository, that can be used as quick references when users are analyzing a certain claim.

2.4. Technologies involved in the process:

While numerous agencies and organizations exist that provide specialised verification or disprofment of news, the analyses that they provide are not in a common consistent way. In order to provide the FANDANGO platform end-user this information in a consistent manner, we gather the content in our own database while at the same time we ensure that data follow our data models.

The data gathering procedure is performed by using multiple spiders to crawl known providers of claim reviews. The spiders are written in the Python programming language and use the scrapy library which allows our setup to be very versatile and robust at the same time.

Unfortunately, only a few of these providers include their analysis in structured data schemas like the one defined by schema.org. A great number of analyses is delivered in text format, which is analyzed by our system in the same manner as news articles' text. This approach allows the system to extract descriptors, comparable between articles and claim reviews.

To analyze the database content and provide the real-time contextual information to users, a web service structure will be created with two different services: similar claims analysis; and similar open-data analysis. These two web services will provide similar functionality, where it performs a relevance analysis over existing records and returns the top five, most relevant, records based on a certain statement. While the fields and data content used on each web service is unique to the object in question, either open data references or claims, the architecture and the analysis done is technologically the same.

These web services are simple REST services implementations, using Node.js, and act as interfaces between the front-end, UI, and the back-end, analysis, that will be done in Elasticsearch. It is also where the adjustable parameters used to fine tune the search and similarity analysis are defined and can be adjusted to increase relevance in certain fields or document characteristics used by Elasticsearch to process the information.

Once a request from the web interface is sent to web services, it them composes the appropriate analysis criteria and requests the Elasticsearch engine to search and rank the results according to the similarity scoring. The best type of similarity analysis will be adjusted according to user feedback, but initial similarity scoring will be done by using LM Jelinek Mercer and Okapi BM25 similarity analysis method, where only the top five results will then be returned by the service.

This process enables FANDANGO to deliver content through a modern information retrieval process, facilitating the adoption of the system when compared to traditional fact checking processes that are hugely reliant on searching for information on web search portals like Google and Bing.

2.5. Process Output:

As a final result of this process, users will be able to see relevant claims, associated reviews of trusted sources and open data references, simply by selecting or typing a statement to be analyzed through the web interface. The main goals in this case are: improving overall user efficiency at finding relevant

fact-checking information, increasing confidence in the analysis process and rising accuracy rates when identifying misinformation.

2.6. Non-functional requirements:

Crawlers for the claim reviews are provided in a docker container, thus allowing for fast redeployment in many systems to parallelize the burden.

3. Image and video verification

Media plays a very important role in news articles. Images and videos are often used as corroboration to the story in an article. It is very important for the Fandango project to be able to automatically evaluate media and mark them as authentic or manipulated. Additionally, even if a media file is authentic, the system should also check if the specific media file coincides with the information existing in the other modalities of the article. In order to make this evaluation, the Fandango system will provide for a number of different descriptors that will help understand the time frame when the media was captured, the location that is depicted and the objects that are displayed.

3.1. Goal:

The image and video verification tools that are being developed for the Fandango project will have two main goals: first to validate if the media is authentic or it has been manipulated in some way, and second to provide image and video descriptors about the where, when and what of media included in an article.

3.2. Precondition (User requirements Addressed):

In the Fandango user requirements (deliverable “D2.3 User Requirements”) it is stated by the users, that fake news cannot be detected by a single indicator, but multiple indicators should be considered simultaneously. These indicators will provide signals at a granular level, where each modality will be independently evaluated, and at the article level where modalities will be evaluated based on the content and evaluation of the indicator by all the modalities present in the article.

3.3. Chosen Approach:

The image and video verification tools aim to provide indicators for both of these levels. For the first, more granular level, we are developing a number of machine learning models that can detect various different attacks on media, like copy-paste, splicing and manipulation for images, and video processing and editing for videos, even deep fakes. Image and video manipulation attacks are not unique in the field of fake news, therefore, a number a publicly available datasets exist that can be used to train and evaluate the provided models. A detailed description on all the developed models will be provided in deliverable “D4.3 Copy-move detection on audio-visual content prototypes”.

For the second level, a more in depth analysis of the media modality has to be performed, in order to provide the end user with content dependant indicators. For this level of indicators, we will answer the “when”, “where” and “what” for a media file.

In order to answer the “what” we are using a deep learning object detection model. This model will initially be trained in generic object datasets (like ImageNet and Ms-coco) but then the model can be fine

tuned with media better suited to the fake news detection field, by annotating media with specific object labels (i.e for better detection of fake news concerning immigration one could use labels like “inflatable boat” and “borders” and train the image models to detect instances of these labels inside an image). The output of this process will be an accounting of how many instances per label are depicted in an image.

A second indicator for the content of an image will try to answer the “where” an image was taken based on the visual content. By using the publicly available Places 2 dataset, we will provide a deep learning model which will return location based markers of what is depicted like “restaurant”, “park” and “seaside”. Additionally, we will refine those labels by also detecting well known landmarks around the world. In order to achieve this we will train a model using the only recently published google landmarks dataset.

Finally, the “when” of a media file will be answered by searching inside the Fandango data lake for similar images already used in older articles. This way the end user will be presented with possible duplicates of the image in question that have already been used in other occasions. By examining the retrieval dates of such images, one can better evaluate when an image was first published.

A detailed explanation of all the models will be presented in deliverable “D4.1 Spatio-temporal analytics and out of context fakeness markers prototypes”.

3.4. Technologies involved in the process:

Current state of the art in both image and video manipulation detection is based on deep learning technologies. Therefore, in the Fandango project a full stack of deep learning libraries will be used in order to offer the best approaches available. Libraries such as tensorflow/keras and pytorch will be used both for the image and video classification models and the various indicators’ models. We will also use robust but lightweight frameworks for the API and user interface like flask and django. Finally, image handling libraries like pillow and opencv are also necessary.

3.5. Process Output:

When validating the authenticity of a media file, the degree of manipulation, while important, is not very useful for understanding what is manipulated. Therefore, instead of returning a single score, our methods will provide the end user with an explanation on what type of attack has been detected and in which part of the media it was detected, i.e a per-pixel probability that an image has been manipulated / tampered with, by each method. Additionally, we return a per method and a total confidence of the models. In order for the end user to validate if an image has already been posted in the past, we return the most similar images that the Fandango system has in the data lake. Additionally, we return the top 5 labels provided by the location module and the top landmark detected, if any. Furthermore, we return the count of all the detected objects.

3.6. Non-functional requirements:

The communication of the front-end, UI, with the analysis modules is done through web services. These web services are simple REST services implementations, using Python and the Flask web framework. Because all the media analyses are slow in their nature, even more so when dealing with video, the services are implemented as asynchronous. The information that is exchanged is in JSON format. The analysis request endpoint expects the media url that is to be analyzed and responds with the media

identifier. The analyzed media endpoint expects the media identifier for the media whose analysis is wanted. The response of the service includes the status of the analysis, the percentage of the analysis that has been performed and the analysis itself which comprises of an overall rating and the prediction of each of the media verification models.

In parallel, for each media item that is inserted to the system for analysis, all the media descriptor models also analyse the image and store the result in the database.

In order to provide for easier deployment all the models are self contained using docker containers and communicate with each other with REST services. We provide versions that work either performing the analysis in the CPU or the GPU. The GPU version can be up to 100x times faster than the CPU during the inference stage.In the current system setup the models use the GPU version on a system providing Titan Xp GPU cards.

OTHER NON-FUNCTIONAL REQUIREMENTS

- Fandango supports its modules/services(described above) in 4 different languages (English, italian, spanish and dutch).

Figure 1 Caption