



ENhancing seCurity and privAcy in the Social wEb: a user-centered approach for the protection of minors



WP6 - Sensitive content detection and protection

Deliverable D6.1: "Design and implementation of in-browser content analysis filter"

Editor(s):	Emanuele Maiorana, Patrizio Campisi (ROMA3)
Author(s):	Rig Das, Himanka Kalita, Emanuela Piciuccio, Gabriel Emile Hine, Emanuele Maiorana, Patrizio Campisi (ROMA3), Loizos Koukoumas, Rafael Constantinou (CyRIC), Antonis Papasava, Michael Sirivianos (CUT)
Dissemination Level:	Public
Nature:	Demonstrator
Version:	0.7









PROPRIETARY RIGHTS STATEMENT

This document contains information, which is proprietary to the ENCISE Consortium. Neither this document nor the information contained herein shall be used, duplicated or communicated by any means to any third party, in whole or in parts, except with prior written consent of the ENCISE consortium.

ENCASE Project Profile

Contract Number	691025
Acronym	ENCASE
Title	ENhancing seCurity and privacy in the Social wEb: a user-centered approach for the protection of minors
Start Date	Jan 1 st , 2016
Duration	48 Months

Partners

	Cyprus University of Technology	Cyprus
	University College London	United Kingdom
	Aristotle University	Greece
	Università degli Studi Roma Tre	Italy
	Telefonica Investigacion Y Desarrollo SA	Spain
	SignalGenerix Ltd	Cyprus
	Cyprus Research and Innovation Center, Ltd	Cyprus
	L S Tech	Spain

Document History

AUTHORS

- (ROMA3) Rig Das, Himanka Kalita, Emanuela Piciuccio, Gabriel Emile Hine, Emanuele Maiorana, Patrizio Campisi
- (CYRIC) Loizos Koukoumas, Rafael Constantinou
- (CUT) Antonis Papasavva, Michael Sirivianos

VERSIONS

Version	Date	Author	Remarks
0.1	20.06.2018	ROMA3, CyRIC	Initial partial draft
0.2	22.06.2018	ROMA3, CyRIC	Partial draft
0.3	26.06.2018	ROMA3, CyRIC	Partial draft
0.4	27.06.2018	ROMA3, CyRIC, CUT	First complete draft
0.5	29.06.2018	ROMA3, CyRIC, CUT	Proposed release
0.6	30.06.2018	ROMA3, CyRIC, CUT	Revised release
0.7	02.07.2018	ROMA3,CUT	Final

Executive Summary

The overall aim of the ENCASE project is to leverage the latest advances in usable security and privacy-preserving technologies, in order to design and implement a browser-based user-centric architecture for the protection of minors (age 10-18) from malicious actors in online social networks.

This deliverable D6.1 - “Design and implementation of in-browser content analysis filter” is realized within the context of the Task T6.1 - “Automatic detection of sensitive personal content”. This Task aims to develop sophisticated content analysis algorithms able to automatically detect contents, including text, images, and videos, that may contain user private data that could eventually pose a risk to the integrity and safety of the user.

The main objectives of the aforementioned Task are:

- a) the generation of annotated training/tests set that would can be used for training/testing the algorithms to be developed;
- b) the determination of key image processing tasks in relation to the detection of private content (e.g., face recognition, expression recognition, body pose recognition, and nudity detection);
- c) the development of algorithms for each of the aforementioned image interpretation tasks;
- d) the development of the computational linguistics and NLP algorithms to analyze the content of the communication between users (messages or newsfeed posts) for evidence of sensitive content (e.g., address information);
- e) and the integration of the developed algorithms in a content analysis filter (CAF) installed on user's browsers, so that an on-line user is warned when content that could potentially contain private data is being uploaded from her browser on an OSN or appears on other users' newsfeeds.

This document describes in detail and demonstrates the progress made towards automatically detecting this threat, with specific emphasis to the algorithms implemented to be included in the aforementioned in-browser filter.

Table of Contents

Executive Summary	4
List of Figures	7
1. Introduction	8
2. Architecture of the proposed in-browser content analysis filter	11
2.1. Filter Architecture	11
2.2. Implementation Stack	11
3. Textual Content Analysis	12
3.1. Stanford Core NLP	14
3.2. Description of our Proposed Algorithm	15
3.3. Examples of Execution of Proposed Algorithm:	16
3.4. Software/Tool Download Link:	21
3.5. How to Install and Run the Code:	21
3.6. Summary	23
4. Automatic Detection of Sensitive Personal Content Using Computer Vision	23
4.1. Datasets Summary	23
4.2. Face Detection Algorithms Review	24
4.2.1. Pre-processing Algorithms	25
4.2.2. Viola and Jones Algorithm	25
4.3. Face Recognition Algorithms Review	27
4.3.1. Eigenfaces	27
4.3.2. Fisherfaces	27
4.3.3. Local Binary Patterns Histograms (LBPH)	28
4.4. Skin Detection Algorithms Review	29
4.4.1. Adaptive thresholds	29
4.4.2. Fusion Approach	29
4.5. Nudity Detection Algorithms Review	30
4.5.1. Scale Invariant Feature Transform (SIFT)	30
4.5.2. Bag of Features (BoF)	31

4.6.	Algorithm Development and Code Recreation.....	31
4.6.1.	Framework Introduction.....	32
4.6.2.	Face Detection and Eye Detection.....	32
4.6.3.	Skin Detection.....	33
4.7.	User Guide	34
4.7.1.	Run the Program.....	34
4.7.2.	Run the Code	34
5.	Conclusion.....	35
6.	Bibliography.....	36

List of Figures

Figure I –Architecture of the proposed in-browser content analysis filter	11
Figure II – General examples of use of CoreNLP	15
Figure III – Example of use of CoreNLP: name detection	17
Figure IV – Example of use of CoreNLP: address detection.....	18
Figure V – Example of use of CoreNLP: sexual abuse detection.....	18
Figure VI – Example of use of CoreNLP: insult detection	19
Figure VII – Example of use of CoreNLP: bullying detection.....	20
Figure VIII – Example of use of CoreNLP: repetitive character and sexual abuse detection.....	20
Figure IX – Example of use of CoreNLP: phone number detection	21
Figure X- Five types of Haar Wavelet-like features	26
Figure XI – The integral image calculations example.....	26
Figure XII – Eigenfaces Algorithm working	27
Figure XIII – Fisherfaces Algorithm Working.....	28
Figure XIV – Local binary distribution	28
Figure XV – The fusion approach combining GM and Smoothing Histogram	30
Figure XVI – Bag of Features recognizing items on an image	31
Figure XVII – Face and Eye detection demonstration.....	32
Figure XVIII – Original image/RGB/HSV/YCbCr and the end average result.....	33

1. Introduction

Nowadays, there is increasing evidence that malicious activities performed through the web, implying cyberbullying, sexual predation, distressed or aggressive behaviour, and more, are actually widespread, and that minors in the age 10-18 are the preferred victims of such kind of acts. These issues have been for long underreported, often leading to dramatic consequences. For instance, episodes of severe school violence have ignited global concern for the safety of school children over the last decade. Such concern has stimulated research about cyber-bullying and numerous other related activities.

Bullying occurs when someone, who is stronger, either physically or emotionally, uses his or her power to intimidate, threaten, or blackmail others. Traditionally, bullying among minors has taken place throughout the school day, and children could retreat to the safety of their homes once the school bell rang. Whereas bullying was once believed to be harmless playground behaviour, it is now known to have long-lasting harmful effects, for both the victim and the bully. Actually, minors who experienced bullying, both as a victim and an offender, had significantly lower self-esteem than those who had little or no experience with it. Bullying impacts students’ wellbeing, schooling, and peer relationships.

With the growth of digital technology, the methods used by bullies have changed dramatically. The use of e-mail, mobile phone text messaging, mobile phone calls, mobile phone cameras, chat rooms, and especially online social networks (OSNs), has taken the concept of bullying to a new, heightened level, i.e., which can occur 24 hours a day, 7 days a week.

A non-inclusive list of the possible scenarios a minor may face nowadays is reported as follows:

- cyber-harassment involves repeatedly sending nasty, mean, and insulting messages; making fun of others or calling them with hurtful names; completely ignoring and excluding others from social groups; telling lies or spreading false rumors for another individual. Harassment by proxy happens when cyberbullies get someone else (or several people) to do their dirty work, or when a bully intentionally provokes a victim to lash back to get the victim in trouble. Harassment through the web may imply also the presence of an adult.

Within this category, we can mention with more detail:

- cyberstalking, consisting in sending abusive messages repeatedly through the Internet or by using a mobile phone. The messages are often threatening in nature, and instil fear that the stalking might move offline and into the target's real life, even becoming physically threatening;

- denigration, implying sending or posting gossip or rumors about a person to damage his or her reputation or friendships
- falsify identity occurs when the offender hacks another's account, and begins posting content or pictures aimed at causing embarrassment or reputation damage to the victim, often resulting in isolating the victim from others;
- flaming refers to arguments or messages sent through electronic communication (chat rooms, e-mail, instant messenger, etc.), supplemented with graphics, specific images, and harsh language to drive home a point. Examples include photo and video postings and sexting;
- masquerading/impersonation is a sophisticated form of cyberbullying in which an individual creates a false identity and harasses another while pretending to be someone else. Masquerading or impersonation can include theft of another person's login information to broadcast harassing or humiliating information about the target online. Pretending to be someone else and sending or posting material to get that person in trouble or danger or to damage that person's reputation or friendships;
- online grooming happens when a predator builds an online relationship with a child by giving compliments, or a "shoulder to lean on", or sending gifts until the child trusts the predator. Typical 'grooming' lines include:
 - a) where is the computer in the house?
 - b) are your parents around much?
 - c) you can always talk to me.
 - d) I'm always here for you.
 - e) you don't deserve how they treat you.
 - f) you have a great personality.
 - g) you're beautiful. You should be a model.
- outing consists in an individual disclosing private information online to friends that is then disseminated over the web through social web sites and/or mobile phones. That person is "outed," with devastating effects. It is often these situations that have led to the teen suicides of late, as these targets do not know how to regain control of their lives after such broad public humiliation;
- phishing is an attempt to get your personal information by pretending to be a site you are familiar with or trust;
- sexting is sending sexually explicit messages via cell phone or instant messenger. As technology has advanced and cell phones have the capability to record and send photos and video, the practice of sending suggestive and explicit pictures has increased especially among teens;

- trickery means talking someone into revealing secrets or embarrassing information or images online.

Due to the variety of scenarios minors can face, and the vastness of potential menaces they should be guarded from, it is very hard for parents and institutions such as schools to control the aforementioned behaviours, with a consequent urgent need for automatic methods able to detect and possibly avoid all the discussed undesired situations.

The design of algorithms able to automatically detect contents, including images and videos, that may contain user private data that could eventually pose a risk to the integrity and safety of the user, is therefore of paramount importance. Specifically, the capability an automatic system protecting minors from possible abuses should comprise:

- the ability of detecting whether personal information are being disclosed;
- the ability of understanding whether seductive or inappropriate pictures of the minor or of others are shared online;
- the ability to check whether e-mails, or chats, or comments shared through social networks contain insults and can be characterized by aggressive behaviour.

The present document describes feasible solutions which could be implemented through an in-browser add-on filter, able to perform the aforementioned tasks and warning both the user, as well as his/her parents or tutors, of possible misconduct which should be avoided or reported to authorities.

In more detail, Section 2 deals with the architecture proposed for the required content filter. Section 3 then specifically considers the scenarios relevant when evaluating text contents sent through chat, instant messaging or social networks. Section 4 instead treats the issues associated with the detection of personal and sensitive content in images or videos. Brief descriptions of the implemented codes are also provided. Eventually, conclusions regarding the performed activities are given in Section 5.

2. Architecture of the proposed in-browser content analysis filter

This section presents the architecture of the designed in-browser add-on for content analysis.

2.1. Filter Architecture

The browser has an installed add-on that collects data. The data is then separated depending on its form, text or image. When the data is sent, it is sent as an API call on either the text filter or the image filter. Depending on the API call, the corresponding algorithm will run behind the scenes and return a result (again depending on the call). That result is then returned to the add-on, which handles the outcome by either filtering or notifying or both. Then, if the data are allowed to be stored, are forwarded and stored accordingly in the database.

2.2. Implementation Stack

Firstly, we have the front-end where in our case could be the add-on or any other website that requires our API. The website in Collaboration with an API manager (this case Swagger) would make an API request. Our current server runs on NGINX with Gunicorn as an App server and Falcon for the APIs. The API will get the required application from the back-end. When back-end finish processing the data, it will return a result and, in some cases even store the data sample.

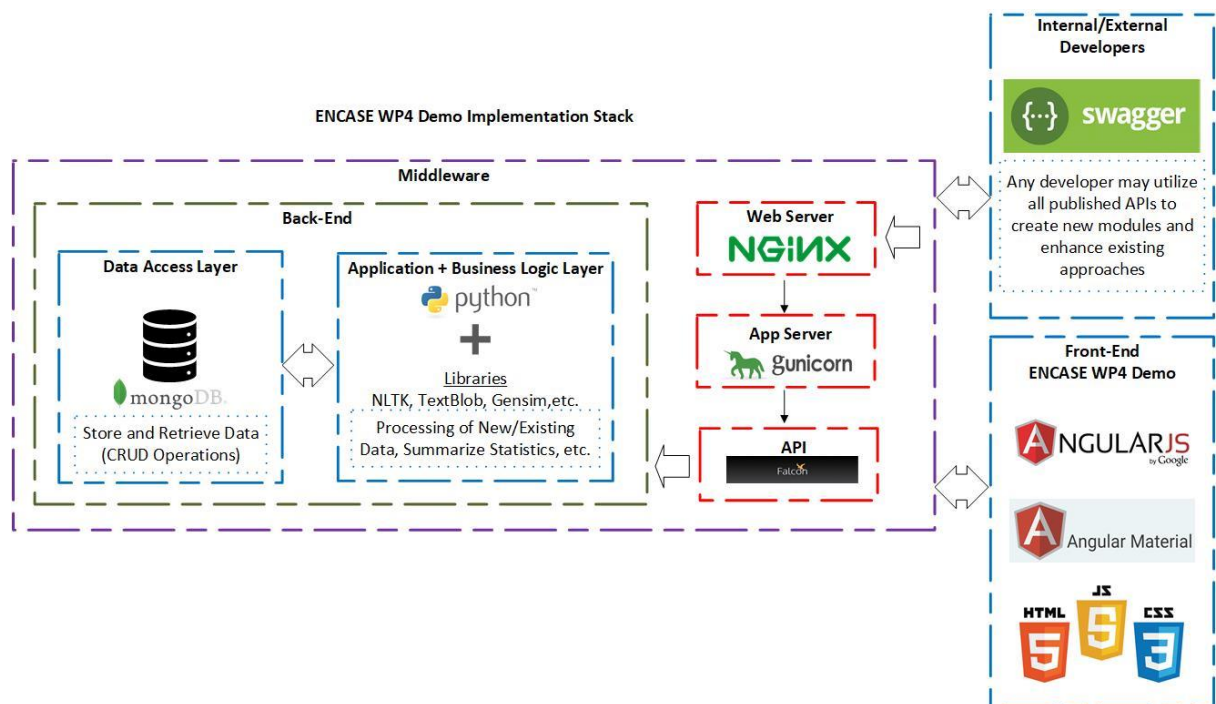


Figure I –Architecture of the proposed in-browser content analysis filter

3. Textual Content Analysis

A part of Task 6.1 states about the involvement of computational linguistics and NLP algorithms to analyse the content of the communication between users (messages or newsfeed posts) for evidence of sensitive content (e.g., address information). This part of the task was the focus of ENCASE secondment #20 from ROMA3 to CyRIC. The target of this secondment was to identify, warn and offer mitigation options for sensitive content shared online.

Before we look into the content analysis section of messages and newsfeed we should carefully review the General Data Protection Regulations (GDRP) of EU, which came into force from 25 May, 2018. According to the GDPR the following categories of sensitive content:

1. Private data: Information groupings that can be listed together or used alone to identify an individual must be stored securely and anonymised (or pseudonymised) accordingly to protect an individual’s identity, such as:
 - name, such as full name, maiden name, mother ‘s maiden name, or alias;
 - personal identification number, such as social security number (SSN), passport number, driver ‘s license number, taxpayer identification number, patient identification number, and financial account or credit card number;
 - address information, such as street address or email address;
 - asset information, such as Internet Protocol (IP) or Media Access Control (MAC) address or other host-specific persistent static identifier that consistently links to a particular person or small, well-defined group of people;
 - telephone numbers, including mobile, business, and personal numbers;
 - personal characteristics, including photographic image (especially of face or other distinguishing characteristic), x-rays, fingerprints, or other biometric image or template data (e.g., retina scan, voice signature, facial geometry);
 - information identifying personally owned property, such as vehicle registration number or title number and related information;
 - information about an individual that is linked or linkable to one of the above (e.g., date of birth, place of birth, race, religion, weight, activities, geographical indicators, employment information, medical information, education information, financial information).

2. Personal data: Personally Identifiable Information, such as:
 - full name (if not common);
 - home address and any other location data;

- email address (if private from an association/club membership, etc.);
- national identification number and any other recognisable ID type field;
- passport number;
- IP address;
- vehicle registration plate number;
- face, fingerprints, handwriting or any other biometric information;
- digital identity;
- date of birth and age;
- birthplace;
- telephone numbers;
- login name, screen name, nickname, or handle.

3. Sensitive personal data:

- racial or ethnic origin;
- political and social opinions;
- religion;
- memberships i.e. trade union;
- health data including physiological and mental;
- sex life;
- medical details and history;
- genetic information;
- banking or other financial details;
- criminal record/activity;
- names and information about parents/siblings/children and/or spouse;
- private assets information.

Based on the above-mentioned types of sensitive data our plan was to build a sensitive content inspector and identifier, which is able to search into the text that is shared online by the user and warn the user regarding the sensitive information that may be containing inside the concerned text. For this purpose, we used already available tools¹ and regular expressions to identify sensitive content both for the EU^{2,3,4} and internationally⁵. The next step of the process will be to

¹ <https://cloud.google.com/dlp/docs/inspecting-text>

² <https://blog.varonis.com/finding-eu-personal-data-with-gdpr-patterns/>

³ <https://community.hds.com/community/products-and-solutions/content-mobility/blog/2016/12/13/gdpr-starts-with-awareness-addressing-article-2a-with-hitachi-content-intelligence>

provide the user with a mitigation strategy that will allow either delete the sensitive content shared or anonymize it⁶.

Based on the above mentioned three types of sensitive data we have planned and organized our sensitive personal content analyzer. Therefore, in order to analyze the message content, we have reviewed some of the state-of-the-art techniques that are publicly available. One of them is [Stanford NLP toolbox](#) [1] and another toolbox for insulting text detection is [Automatic detection of insulting text](#). We have looked into these two types of tools as an example and also as a sample to proceed for our own personal content analyzer.

3.1. Stanford Core NLP

Stanford CoreNLP [1] provides a set of human language technology tools. It can give the base forms of words, their parts of speech, whether they are names of companies, people, etc., normalize dates, times, and numeric quantities, mark up the structure of sentences in terms of phrases and syntactic dependencies, indicate which noun phrases refer to the same entities, indicate sentiment, extract particular or open-class relations between entity mentions, get the quotes people said, etc.

Stanford Core NLP comprises:

- an integrated NLP toolkit with a broad range of grammatical analysis tools;
- a fast, robust annotator for arbitrary texts, widely used in production;
- a modern, regularly updated package, with the overall highest quality text analytics;
- support for a number of major (human) languages;
- available APIs for most major modern programming languages;
- ability to run as a simple web service.

Stanford CoreNLP’s goal is to make it very easy to apply a bunch of linguistic analysis tools to a piece of text. A tool pipeline can be run on a piece of plain text with just two lines of code. CoreNLP is designed to be highly flexible and extensible. With a single option you can change which tools should be enabled and disabled. Stanford CoreNLP integrates many of Stanford’s NLP tools, including the part-of-speech (POS) tagger, the named entity recognizer (NER), the parser, the coreference resolution system, sentiment analysis, bootstrapped pattern learning, and the

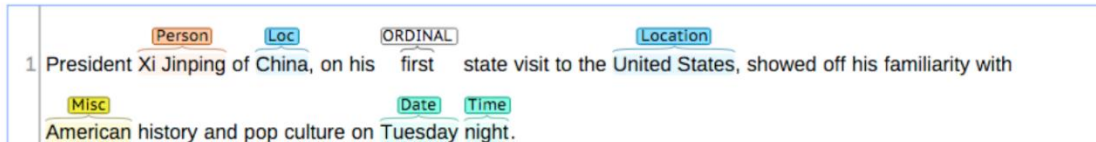
⁴ <http://ieeexplore.ieee.org/document/8115578/>

⁵ <https://technet.microsoft.com/en-us/library/jj150541%28v=exchg.160%29.aspx?f=255&MSPPErr=-2147217396>

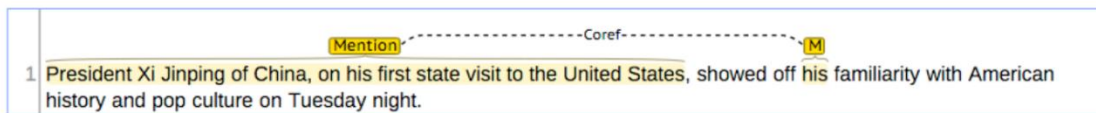
⁶ <https://docs.splunk.com/Documentation/Splunk/7.0.2/Data/Anonymizedata>

open information extraction tools. Moreover, an annotator pipeline can include additional custom or third-party annotators. CoreNLP’s analyses provide the foundational building blocks for higher-level and domain-specific text understanding applications. Following are some of the examples of use of CoreNLP.

Named Entity Recognition:



Coreference:



Basic Dependencies:

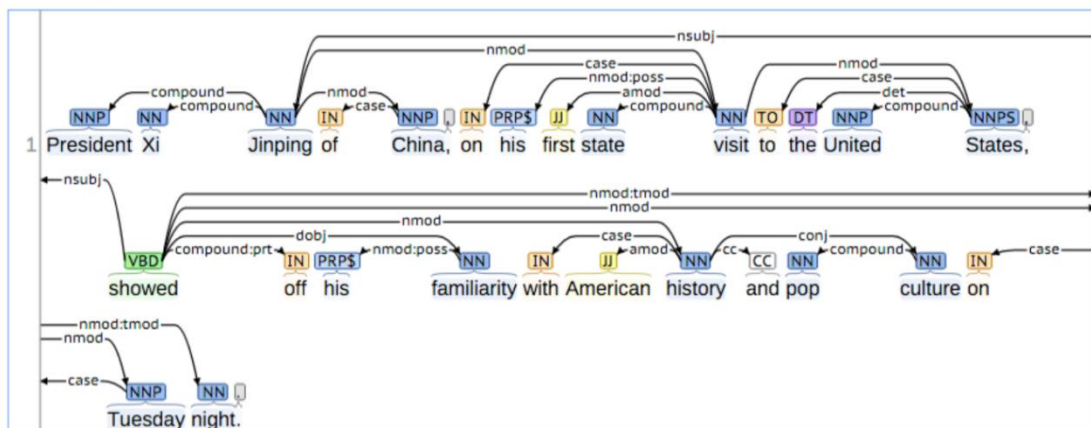


Figure II – General examples of use of CoreNLP

3.2. Description of our Proposed Algorithm

As discussed above we have used the Stanford Core NLP tool for our proposed algorithm. We have created a “Maven” project using “IntelliJ IDEA 2018.1.3” development toolkit. The toolkit is an open source SDK. The programs for our proposed algorithm are coded in JAVA. Following is the details of our proposed algorithm.

- Step-1: Input the sentence or text that needs to be verified;
- Step-2: Use Stanford Core NLP for extracting the individual words and its textual annotations (Word), extract Part-of-Speech (POS) and extract Named Entity (NE);

- Step-3: If a sentence contains with words with repetitive words then the program transforms those words into its original words;
- Step-4: Based on the Word, POS and NE switch cases are designed for identifying bullying, insult and sexual abuse, name, address, phone number etc;
- Step-5: Based on the results obtained in Step-4 a final statement is printed/displayed, stating the outcome of the input sentences.

For the identification of bullying, insult and sexual abuse we have created 3 separate databases with varieties of words. The databases are expandable and are in ".csv" format.

Our proposed algorithm serves the following purposes:

1. detects cyber-bullying (checks with our created database for the cyber-bullying words and if it exists then responds as the sentence contains cyber-bullying);
2. detects insult (checks with our created database for insulting words and if it exists then responds as the sentence contains insult);
3. detects sexual abuse (checks with our created database for sexual-abuse words and if it exists then responds as the sentence contains sexual abusiveness);
4. detects if a name of any person is mentioned or shared (according to GDPR now sharing someone's name is also not allowed; so the code detects the name and informs that there is someone's name that is getting shared in a sentence);
5. detects numbers such as PIN Number, social security number, phone number, street number etc;
6. detects any address if it is getting shared. This is again compatible with GDPR;
7. if you write abusive sentences such as "FFFFFFfUuUUuuuUccccCCcKkKKkkk yyyyyYYoOUu" the program converts it to "fuck you" and provides a decision that the words are sexually abusive or insult or bullying.

3.3. Examples of Execution of Proposed Algorithm:

Following are few examples with their screenshots that shows the performance of our code:

1. **Detection of Name of a Person:**

Input Sentence: Peter lives in Rome.

Output: The proposed algorithm detects that Peter is a name of a person using the Stanford NLP lexical analysis.

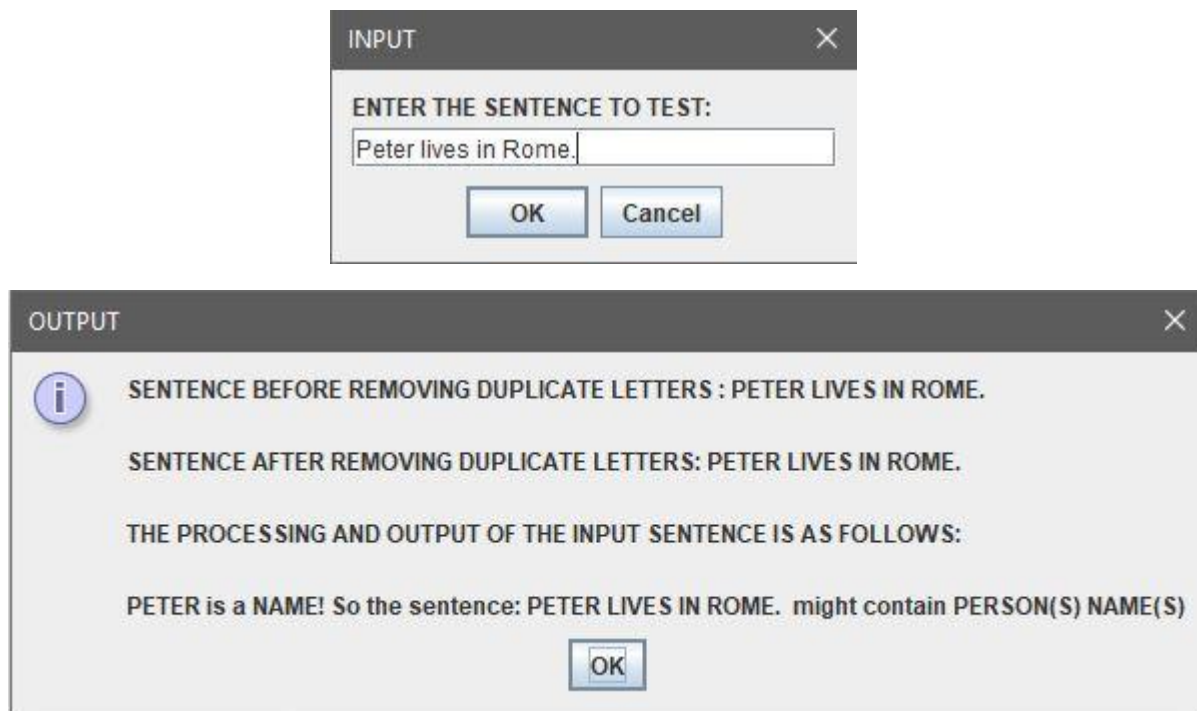
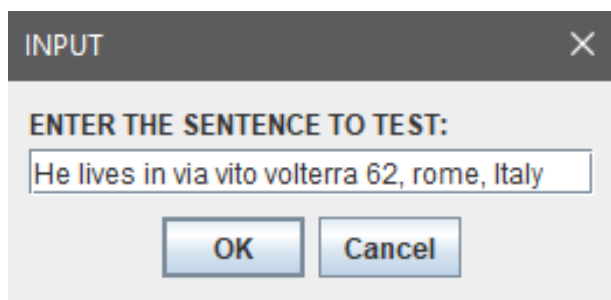


Figure III – Example of use of CoreNLP: name detection

2. Address Detection:

Input: He lives in via vito volterra 62, rome, Italy

Output: The system detects that there is an address that is getting shared again using the Stanford NLP. It also detects that there is Number that is getting shared which might be the street number.



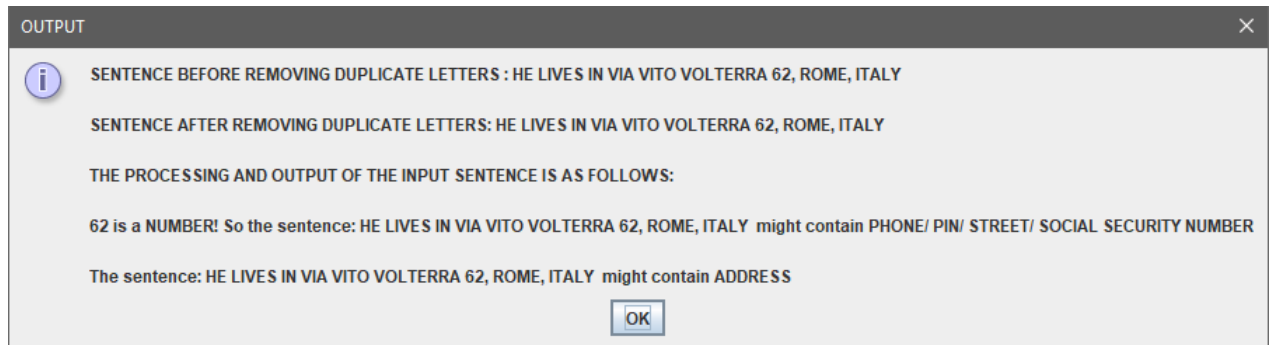


Figure IV – Example of use of CoreNLP: address detection

3. Sexual Abusive Sentence:

Input: You motherfucker

Output: Motherfucker is a sexually abusive word and the sentence is classified as an abusive sentence.

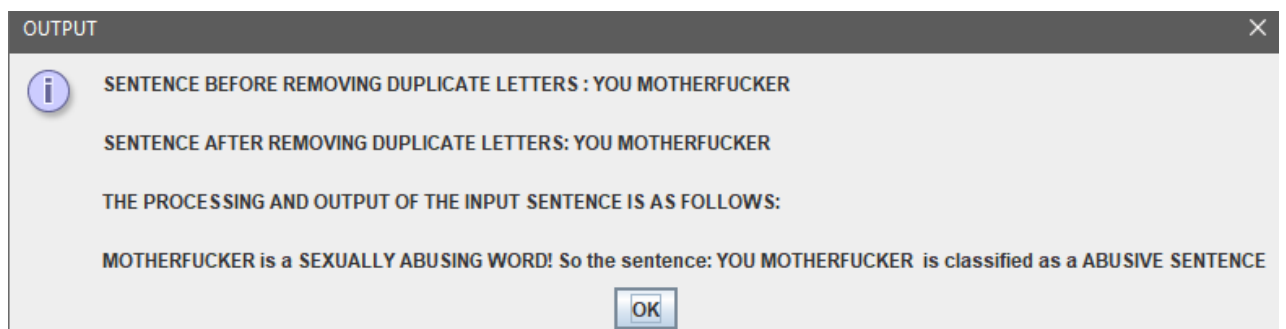
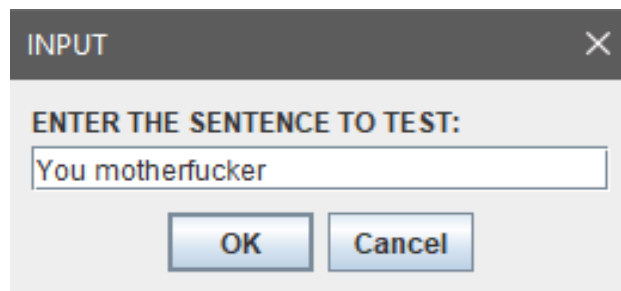


Figure V – Example of use of CoreNLP: sexual abuse detection

4. Insulting Statements:

Input: Go Kill yourself!

Output: This sentence is considered as an insulting statement.

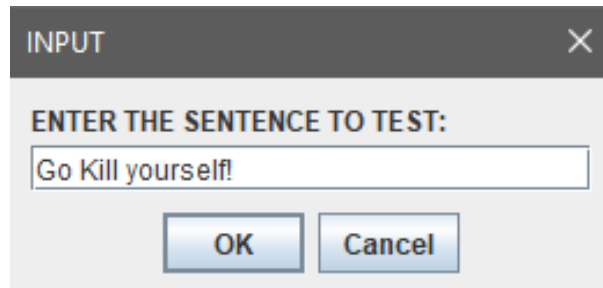
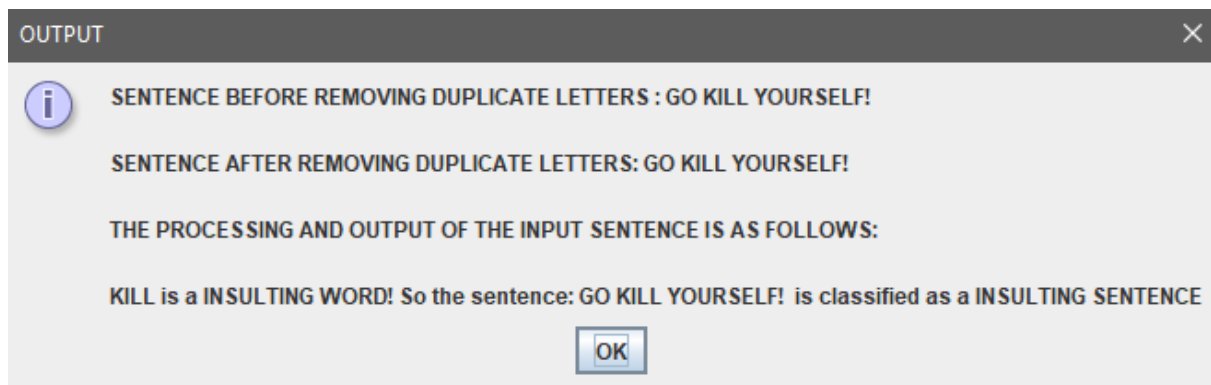
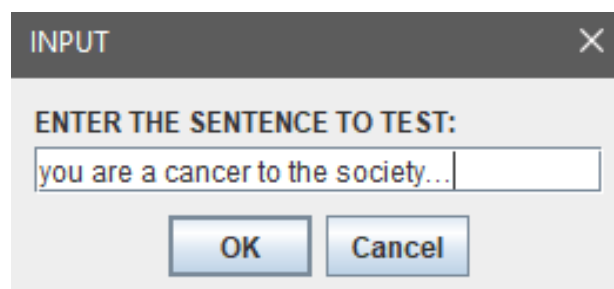



Figure VI – Example of use of CoreNLP: insult detection

5. Bullying Statement:

Input: you are a cancer to the society...

Output: The sentence is considered as a bullying sentence based on the work “cancer”.



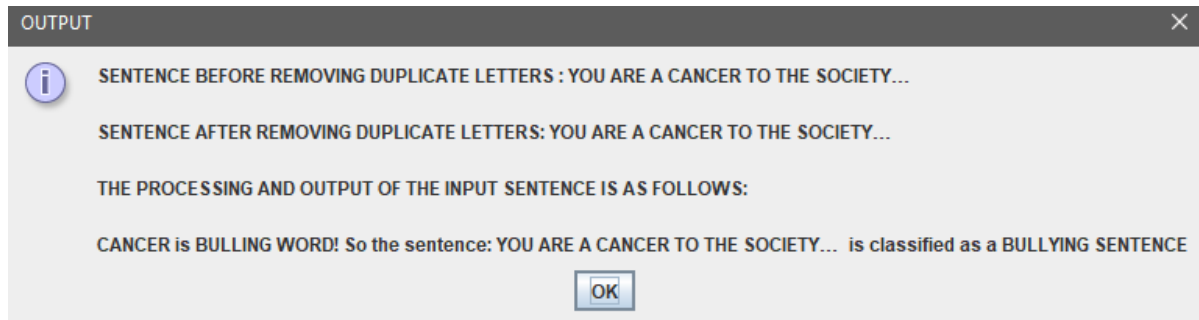


Figure VII – Example of use of CoreNLP: bullying detection

6. **Words with repetitive characters are converted into a proper word and checked through our tool:**

Input: fffffffFuuuuUUUUcCcCcCkkkkk yyou...

Output: The word of the sentence is converted into “FUCK YOU” and when they are checked with our databases it shows that “Fuck is a sexual abusive word” and the sentence is categorized as Sexually abusive.

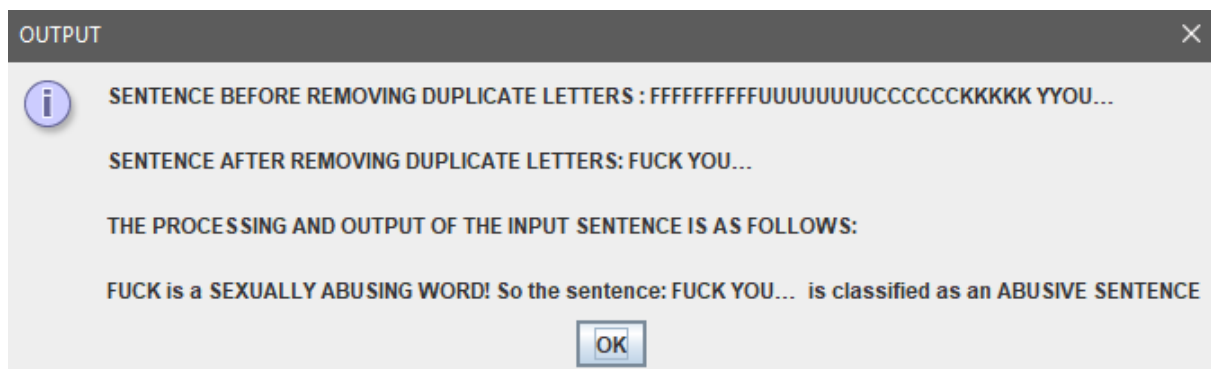
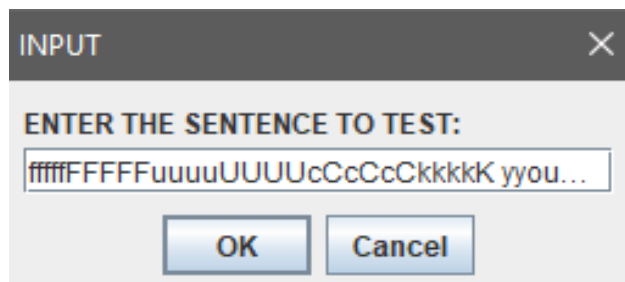


Figure VIII – Example of use of CoreNLP: repetitive character and sexual abuse detection

7. **Phone Number:**

Input: his number is 3411778903

Output: “3411778903” is a Number. The sentence might contain a Phone/ PIN/ Street/ Social Security Number.

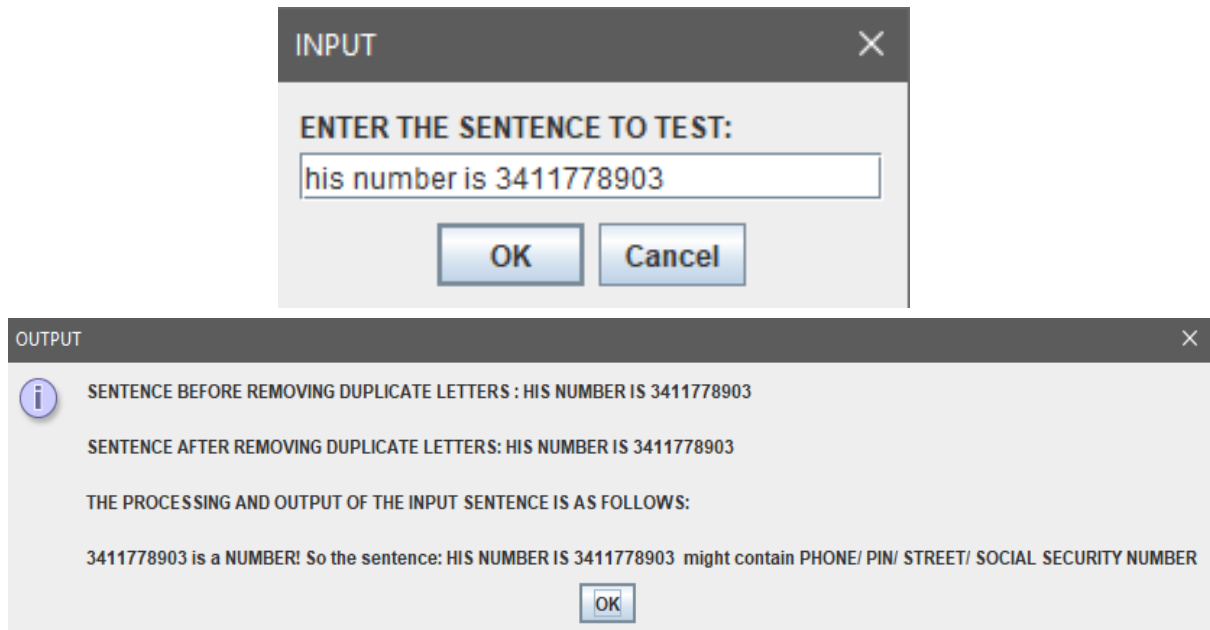


Figure IX – Example of use of CoreNLP: phone number detection

3.4. Software/Tool Download Link:

Please click on the following link for downloading the tool. It is recommended to use Chrome browser to download the file as the file is in ‘.rar’ format and sometimes with other browser the file doesn’t get downloaded properly.

https://www.dropbox.com/s/ei9jp0ygvdc3o0/Core_NLP_Example_2.rar?dl=0

3.5. How to Install and Run the Code:

Extract the “Core_NLP_Example_2.jar” file, “OutFile.txt” file and the “README.txt” file, from the “Core_NLP_Example_2.rar” file provided together with this deliverable, in the exact location given below:

C:/Core_NLP_Example_2/Core_NLP_Example_2.jar

C:/Core_NLP_Example_2/OutFile.txt

C:/Core_NLP_Example_2/README.txt

Three files will be extracted. the **path** is marked in **green**. the **files** are marked in **red**.

Please note that Java SDK is required to run the code. In case JDK is not installed, please install it by checking this [link](#) and downloading the standard edition. It may be required to have also JetBrains IntelliJ IDEA community version. If it is not installed, please check this [link](#) and download the community edition. Please install JDK before installing JetBrains IntelliJ IDEA.

Once all of this has been done, follow any one of the three section’s steps to execute the JAR file:

1. For running in IntelliJ IDEA:

- Open IntelliJ IDEA.
- Go to RUN menu.
- Click on Edit Configurations.
- A dialog box will appear, click "+" sign on left hand top side.
- Select JAR application. On the right side, path to JAR will appear. Browse for JAR file location.
- JRE path is optional, by default it takes the JRE from the installation of JDK. Click Apply. Then click OK.
- Then the dialog box will close.
- You will notice that the Run command will automatically take the JAR file.
- Click on RUN and test the JAR application.

2. For Running from Command Prompt:

- Open command prompt from Start menu (For Windows 10. **Click Start > Windows System > Command Prompt**).
- Type the following command: **“cd C:/Core_NLP_Example_2/”** to go to the directory of your ".jar" file.
- Type the following command: **“java -jar Core_NLP_Example_2.jar”**.
- Press Enter to RUN and test the JAR application.

3. For running by Double Clicking: (Depends on the environment of the operating system where the JAR file is executed)

- Go to the location **“C:/Core_NLP_Example_2/”** from Windows Explorer and double click on the **“Core_NLP_Example_2.jar”** application file.
- The application will run and give appropriate input in the dialog box that will appear.

3.6. Summary

As mentioned in Section 3.2, the three database contains very few but most commonly used words. In order to make the tool more comprehensive the database need to be expanded with more words which will make tool more robust against any kind of communication texts. We suggest that the future programmers and researchers should expand all the three databases for better detection of these three types of sentences. For any further information Roma Tre University can be contacted in future.

4. Automatic Detection of Sensitive Personal Content Using Computer Vision

In addition to the detection of cyber-bullying behavior performed against minors by means of text messages, the carried out activities have also focused on the detection of sensitive content shared through images on the web. Specifically, the presence of faces, skin or nudity contents has been considered as possible issues, and proper algorithms have been developed to deal with these possible scenarios.

This section first presents a brief description of the databases considered to perform the experimental tests carried out to evaluate the effectiveness of the proposed methods. A review of the main face detection and recognition algorithms is the provided, followed by details regarding how to detect skin and nudity contents. The implemented algorithms are then described.

4.1. Datasets Summary

The following is a list of the main databases publicly-available on the web to test applications dealing with face images:

- facial recognition technology (FERET): This database was collected at “George Mason” University and the US Army Research Laboratory facilities. The database documentation lists 24 different facial image categories. Currently the database used 1199 subjects on 2 different expressions, 2 different illuminations 9 – 20 different poses on 2 different time intervals. The database counts 14, 051 images [2];
- Montreal set of facial displays of emotion (MSFDE): This dataset portrays six different emotions from 12 different participants from three different continents (Europe, Asia, Africa). Each expression was taken using a direct facial action task and were FACS coded

(Facial Action Coding System). Each expression was then morphed into 5 different levels of intensity [2];

- the Yale face database: The dataset contains 165 greyscale images of 15 individuals. Everyone posed with and without glasses, posed on 3 different types of illumination and with 6 different facial expressions [2];
- the extended Yale face database: Based on the principle that the “Yale Face Database” needed an improvement, this database includes photographs from 10 different subjects. Each subject posed with 9 different ways and with 64 different illumination conditions;
- Taiwanese facial expression database (TFEID): This dataset is consisted by images taken from 40 individuals, on 8 different facial expressions. Each expression has two different intensities (high and low) and two viewing angles (0° and 45°) [2];
- CMU face images dataset: This dataset contains 640 black and white facial images, captured from 20 subjects. Each subject has images of 4 different poses, 4 different expressions, 2 different configurations (sunglasses or not), and size [2];
- the Cohn-Kanade dataset (CK): Includes of 486 images from 97 subjects. The images were taken in a sequence. Each sequence starts with a normal facial expression and ends with a peak. All peak expressions are FACS coded and include an emotion label describing the emotion requested from the subject;
- the extended Cohn-Kanade Dataset (CK+): This dataset is based on the CK dataset. However, this dataset other than larger, it also has labels for the emotions which in contrast with its predecessor are validated based on what they show.

4.2. Face Detection Algorithms Review

Face Detection is the first step on this project. Its reliability it's a major factor on the performance and usability of the entire face recognition system and an important influence for the rest of the work done, that engages with the detection of a human. When provided with a single image the ideal face detector should be able to identify all present faces regardless of their position, scale, orientation age and expression. Moreover, the detector should be able to identify faces regardless of extraneous illumination conditions or the images contents. In our project, the proxy filter needs to be able to detect possible persons in images hence the face detection is necessary. Furthermore, it needs to be able to verify if the detected face belongs to the person monitored.

4.2.1.Pre-processing Algorithms

Image pre-processing is a collection of algorithms that enhance or disenchant an image. The purpose of pre-processing it's to improve the image data, by suppressing unwanted distortions or enhancing some important features that can be useful for further processing [3] [4] [5].

- skin color filtering: Human skin has a unique color distribution from other non-skin objects. Used as a filter, it can obtain candidate region of faces and it might also be used as a skin color detector for special environments [3] [5];
- image normalization: normalization of pixel intensity, helps correct variations of imaging parameters in cameras as well as changes in illumination conditions. The normalization process uses intensity normalization operations like mean value normalization, histogram equalization and illumination correction [3];
- Gaussian mixture Modeling: Face and non-face sub window distributions in high dimensional space are complex. Single Gaussian cannot handle explaining all variations. Therefore, it was found [Ref] that to deal with this type of complexity a different approach was required. The solution proposed is to divide face training data into several face clusters, and non-face training data into several non-face clusters. The number of clusters would be decided empirically. To perform the cluster Mahalanobis distance is used in the image space or some other space. There could be further modeled by its principal components using PCA technique [3].

4.2.2.Viola and Jones Algorithm

Paul Viola and Michael Jones introduced in 2001, the first object detection framework that provides competitive object detection rates in real time [3]. Main characteristics and tools used within this algorithm are described in the following:

- AdaBoost: AdaBoost or Adaptive Boost is a machine learning approach that is used as a classifier. As a classifier algorithm, AdaBoost is used when there is a large amount of data that has to be divided in different categories. AdaBoost is consisting of many different “weak algorithms” (they don't have much accuracy, so it is hard for them to learn) combined who can easier tackle a high dimensional task. It is called adaptive, because depending on the correct result each algorithm has, the algorithm is assigned a corresponding weight. AdaBoost is often used with other algorithms since it helps them improve their results, and that is where ‘Boost’ comes from [6] [3] [4];

- Haar feature selection: Haar feature selection is an object detection algorithm proposed by Viola and Jones. It is a machine learning approach where a cascade function is trained from a lot of positive and negative images. Next, machine is tested on detecting object in other images. Since this can work so good on objects, it is expected that it can also work on faces and thus is part of the Viola Jones face detection framework. The algorithm is using a number of basic types of scalar features (Fig 3.1) for face detection. Features are scalars, calculated by summing up the pixels in the white region and subtracting those in the dark region [2][3][5].

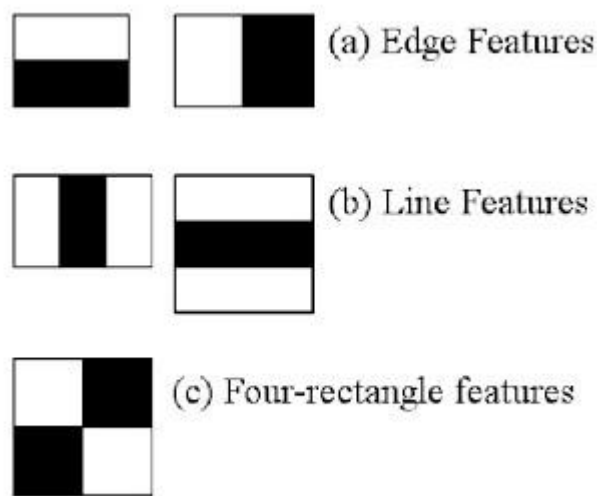


Figure X- Five types of Haar Wavelet-like features

- integral image: Integral image computes the value at each pixel; which is the sum of the pixel values above and to the left of that pixel including its own value. This algorithm is used for calculating the sum of values in a given image. Moreover, it can be used for calculating the average intensity within a given image [2].

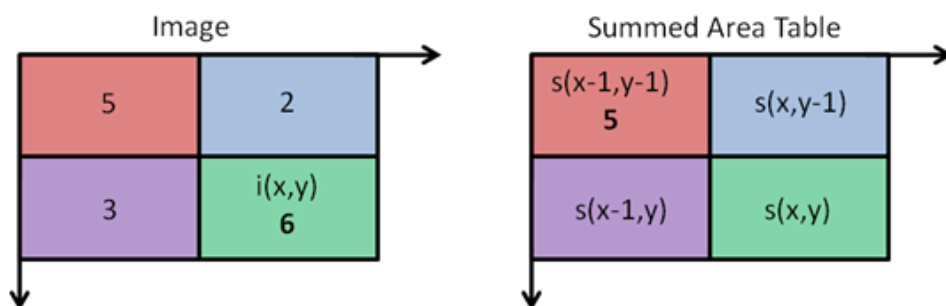


Figure XI – The integral image calculations example

4.3. Face Recognition Algorithms Review

Face recognition is the biometric technology that tries to establish an individual’s identity by storing the individual’s facial features mathematically and stores the data as a faceprint. There are many types of algorithms that are used for this purpose; mostly deep learning and machine learning. After detecting the face in an image, the next step is to verify if that face belongs to the monitored person. The scenario includes that the user of the proxy, would include pictures of the monitored subject, so that the face recognition algorithms can train.

4.3.1. Eigenfaces

Eigenfaces is an algorithm that performs the face recognition by comparing the face to be recognized with some training dataset of known faces. During the training set, we supply the algorithm with faces and the name that the face corresponds to. When the algorithm is asked to recognize some unknown face, it uses the training set to make the recognition. This is a similar way of how Fisherfaces and LBPH works. Eigenfaces however creates a small dataset of only the most meaningful components that account most of the information. That is achieved using the Principal Component Analysis (PCA) [7] [3].



Figure XII – Eigenfaces Algorithm working

4.3.2. Fisherfaces

Fisherfaces algorithm works in a very similar way as Eigenfaces algorithm. However, Fisherfaces considers many of the useful components that might be thrown away from the PCA; The core of the Eigenfaces algorithm. Therefore, in a scenario where the variance in the components come from an external source (light for example), the components identified by a PCA might not contain any discriminative information, therefore the projected samples are mashed together thought they might not contain any discriminative information. Linear Discriminant Analysis (LDA) was invented by *Sir R. A. Fisher* and it’s the core of the Fisherfaces algorithm. What LDA does is to

cluster same classes tightly together, while different classes are kept as far away as possible from each other in the lower-dimensional representation [4] [8].



Figure XIII – Fisherfaces Algorithm Working

4.3.3. Local Binary Patterns Histograms (LBPH)

Local Binary Patterns Histograms algorithm uses the training set as Fisherfaces and Eigenfaces but with a different approach. Instead of creating a mathematical description of the most dominant features in the training set as Eigenfaces and Fisherfaces do, LBPH analyses each face in the training set separately and independently. LBPH, summarizes the local structure of an image by comparing each pixel with its neighborhood. It takes a pixel as a center and threshold its neighbors against it. In case the intensity of the center pixel is greater-equal its neighbor, it denotes it as 1 else as 0. This will lead in a binary number for each pixel (11001111 Fig 4.1). This is the Local Binary Pattern (LBP). Next step for the algorithm is to divide the LBP image into local regions and calculate the histogram from each. These concentrated histograms are called Local Binary Patterns Histogram [3] [9].

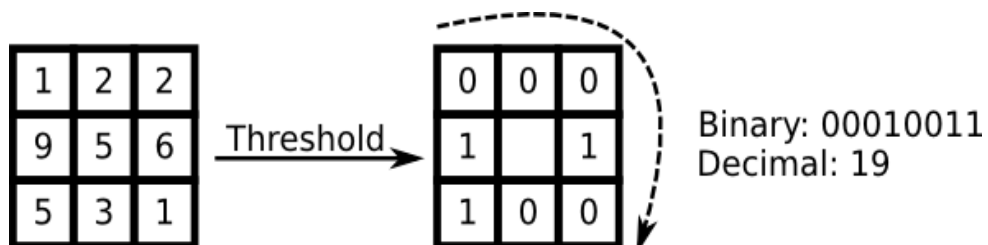


Figure XIV – Local binary distribution

4.4. Skin Detection Algorithms Review

Skin detection is the computer vision process of detecting skin-colored pixels in videos and images. The most common and simplest skin detection algorithms work on defining a fixed decision boundary for different color space components. The typical skin detection algorithm, transforms given pixels into an appropriate color space and then, using a skin classifier, it labels the pixel whether it is a skin or a non-skin pixel.

The main purpose of these algorithms in this project, is to help nudity detection algorithms specify where the skin is, and where to detect nudity.

4.4.1. Adaptive thresholds

The most basic type of skin detection is the color-based detection. There are many color spaces used these days by the computer vision community. The most common ones are: RGB, HSV, YCbCr. Due to its non-uniform and device dependent nature, RGB is usually avoided for colored based analysis. Nevertheless, by using these different color spaces, different filters and thresholds can be applied, that express different results. Therefore, in this approach, RGB was also included. HSV is better suited on simple images with uniform background and YCbCr shines better on situations where there is uneven illumination. Therefore, with combining these results, we get a result with a better summary [10] [11].

4.4.2. Fusion Approach

Fusion approach is a quite revolutionary approach, in terms of using two algorithm results; The smoothed 2D histogram and Gaussian Model. Firstly, an online dynamic approach is employed which excludes the need for a training stage. Then, a 2D histogram with smoothed densities and a Gaussian model are used to model skin and non-skin distributions respectively. Lastly, a fusion strategy framework that uses the product of two features is employed for performing automatic skin detection [12].

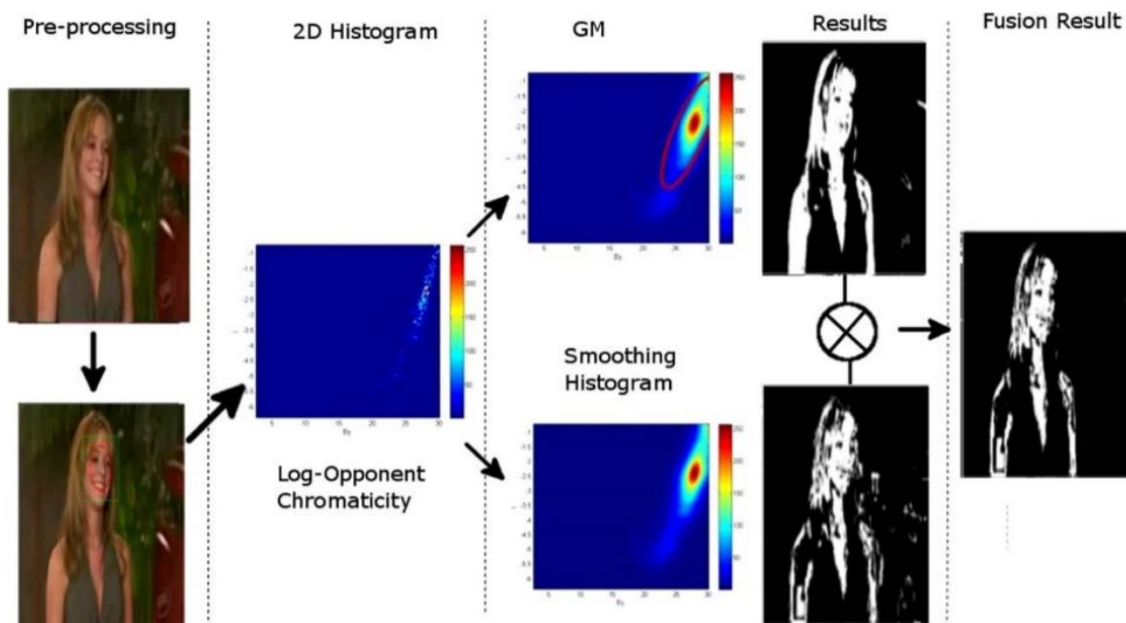


Figure XV – The fusion approach combining GM and Smoothing Histogram

4.5. Nudity Detection Algorithms Review

Nudity detection is also a relevant part of the planned content analysis filter. In the following, the main approaches currently developed for dealing with this scenario are described.

4.5.1. Scale Invariant Feature Transform (SIFT)

The scale invariant feature transform is a computer vision algorithm that specializes on detecting and describing features. SIFT extracts features and describes them regarding their scale, illumination and rotation invariant descriptor. The algorithm follows two main steps. First it detects interest points in an image. Interest points can be anywhere that the 2D signal has variation that exceeds a threshold limit and is superior the simple edge detection. The second step is a process that creates a vector like descriptor. This step is also what makes sift so unique and superior regarding similar algorithms. The scale invariance is created by scanning the interest points at a wide range of scales including the scale of where the interest point resides, i.e. the 2D space wise signal variances. That way when the same process runs on a possible interest point for recognition, it will get the same description even if is in a different actual scale. Then the scale with the most stable presence will be coded. Next step, the same interest points are evaluated on their direction. This is regarding their 2D direction in which the signal variance features have the

highest variability. Consequently, when the direction is known it will be eliminated and it will not obscure direction [13] [14].

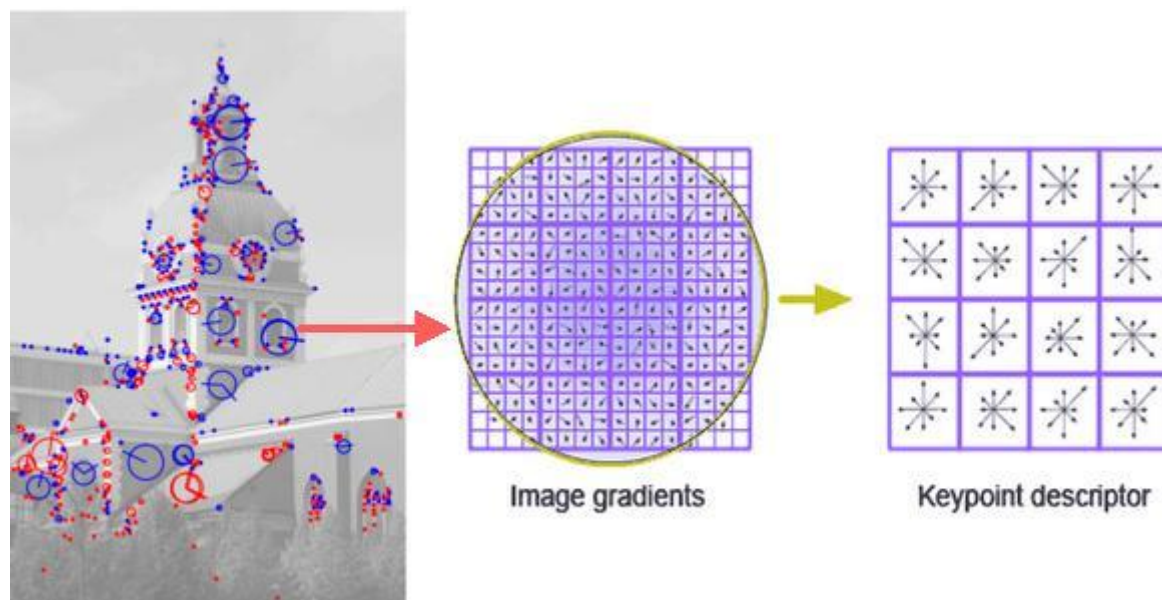


Figure XVI – Bag of Features recognizing items on an image

4.5.2. Bag of Features (BoF)

Bag of Features is a visual descriptor, used for visual data classification. As a concept, it is inspired by Bag of Words which is an algorithm used for document classification. Bag of words we call a sparse vector of word frequency; which is, a sparse histogram over the vocabulary. In BoF case a bag of features is a thin vector of frequency of a vocabulary of local image features. BoF approach is usually involving two main steps. The first step is to obtain the set of bag features. This is usually done offline. On the second step we cluster the set of given features into the set of bags that was created in the first step, and then create the histogram taking the bags as the bins. That histogram is then used to classify the image [12].

4.6. Algorithm Development and Code Recreation

A code is provided together with this deliverable, in order to provide the source information to be integrated into the planned in-browser filter, according to the architecture described in Section 2. A demo example is also provided in order to show the capability of the designed algorithms.

4.6.1. Framework Introduction

The goal of the software development part was to create a “filter” that would be able to detect the face of a subject in an image and if the image contains any sensitive content such as nudity. The programming language used was Python 2.7 along with the OpenCV framework. OpenCV has many algorithms already prepared. Moreover, it contains machine learning scripts that can help speed up the process.

4.6.2. Face Detection and Eye Detection

For improved face recognition, these two algorithms have been merged on a standalone program. The future goal is to be integrated with the current face recognition algorithm.

The program takes as input a file link; a location on the system that contains an image. After the image is loaded, we load the cascade classifier XML file. This file contains a boosted Haar cascade classifier. This file is the result of a training data collection and can be replaced by better ones in case a training produces better results. This version contains one file for the face detection and one for the eye detection. Next, the program indicates where the face and the eyes are and returns the amount of heads and pair of eyes found.

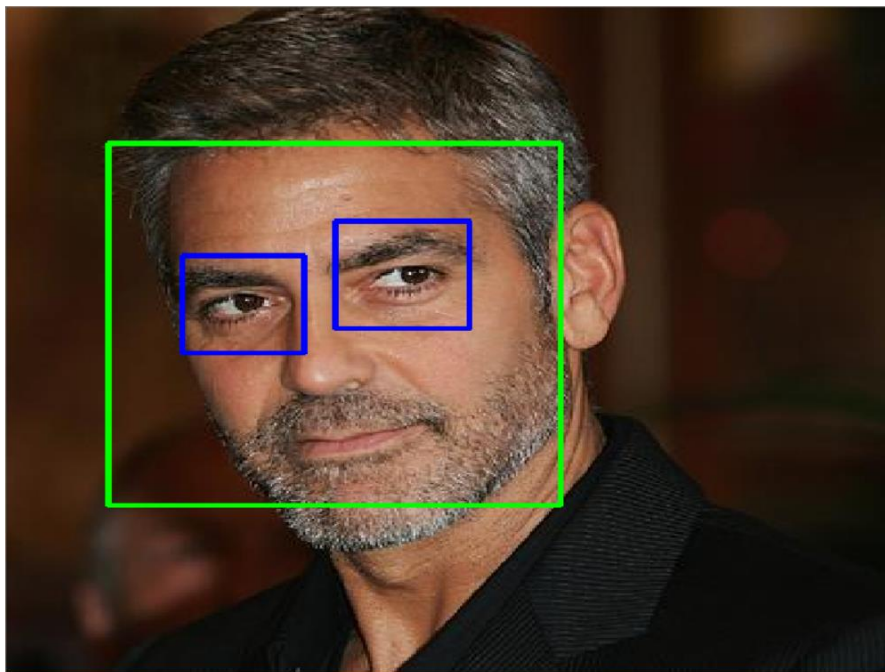


Figure XVII – Face and Eye detection demonstration

4.6.3. Skin Detection

Skin detections main purpose in this project was to help identify nudity. In the end the percentage of nudity in the focus of the image is what is going to help determine nudity. For this purpose, the approach followed was to assign thresholds on three main color spaces (RGB, HSV, YCbCr) [10] [11]. The program takes the image, and then it creates copies on each of the color spaces. Then a threshold mask is applied in the images, removing non skin. Then the average of the masks is combined to create the final result. From Figure XVIII, we can see that some of the masks are better at detecting skin on different situations either due to the skin color or to the illumination conditions.

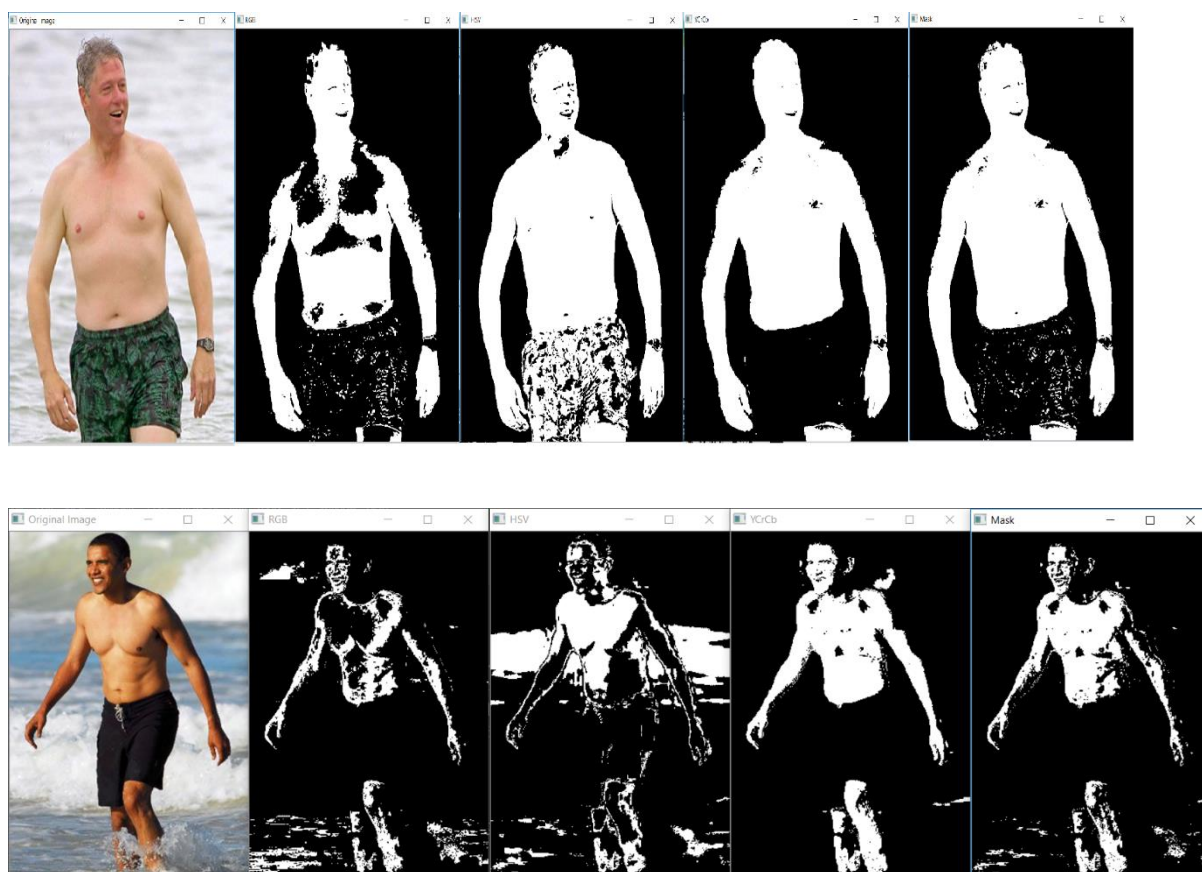


Figure XVIII – Original image/RGB/HSV/YCbCr and the end average result

4.7. User Guide

A file named “Encase_ImageAnalysis.zip” containing the code required to recreate the designed program, as well as the build of the program, is provided together with this deliverable. There is also a folder with data that can be used as a test for the program or for further development from the user.

4.7.1. Run the Program

The program’s executable file location is at `...\Encase\dist\main.exe`. The user simply needs to run the .exe file without any prerequisites. When the program starts, a command prompt window opens, containing the programs menu. The user can navigate through the menu by typing the corresponding number and then pressing enter.

When running for “Face and Eye Detection Mode” or “Skin Detection Mode”, the next step will ask you to specify a .jpg file. Here you can use the images found in the data folder located at: `...\Encase\dist\data\...`

When running the Face Recognition Mode, the program will ask to provide it with a folder that contains the set of image that the recognition will be trained and then tested on. As for the current version there are the following rules that your dataset needs to follow:

- the names of the images need to include the subject number +. something. An example can be found in the “yalefaces” folder on: `“Encase\dist\data\yalefaces”`.
- the testing image needs to end with .test (as also can be seen in the “yalefaces” folder)

4.7.2. Run the Code

For running the code, first you need to have installed and added in the system environments the Python 2.7+ (not 3+) and OpenCV on your computer. Next, it is required that you install numpy 1.7.1+ and matplotlib-1.3.0. To run from console, simply run the main.py file. That should initiate the program. Same goes in case you are running from an IDE. Running/Debugging the main.py file should start the program. The instructions for the interface are the same as when running the program from the executable.

5. Conclusion

The present document provides the description of the activities carried out during the lifetime of T6.1 of the ENCASE project. The building blocks needed to develop an in-browser content analysis filter, able to detect possible malicious behavior implying cyber-bullying, have been described here. The performed research has been focused on the design of algorithm able to evaluate whether generic text messages contain aggressive or insulting content, or express sensitive information which should be preserved in order to protect users’ privacy. The issues associated with the sharing of image content have been also taken into account, developing an application able to detect whether face, skin, or nudity content are downloaded or uploaded to the web.

6. Bibliography

- [1] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard e D. McClosky, «The Stanford CoreNLP Natural Language Processing Toolkit,,» in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistic: System Demonstrations*, 2014.
- [2] K. Kaulard, D. W. Cunningham, H. H. Bülthoff e C. Wallraven, «The MPI Facial Expression Database — A Validated Database of Emotional and Conversational Facial Expressions,» *PlosOne*, 2012.
- [3] S. Z. Li e A. K. Jain, *Handbook of Face Recognition*, New York City: Springer, 2004.
- [4] E. Arubas, «Face Detection and Recognition (Theory and Practice),» 6 April 2013. [Online]. Available: <http://eyalarubas.com/face-detection-and-recognition.html>.
- [5] N. H. Barnouti, «Improve Face Recognition Rate Using Different Image Pre-Processing Techniques,» *American Journal of Engineering Research*, vol. 5, n. 4, pp. 44-53, 2016.
- [6] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mohabi e Y. Ma, «Towards a Practical Face Recognition System:», *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, n. 2, pp. 372-386, 09 March 2012.
- [7] S. Zhang e M. Turk, «Eigenfaces,» *Scholarpedia*, 18 July 2008. [Online]. [Consultato il giorno March 2018].
- [8] A. Martinez, «Fisherfaces,» *Scholarpedia*, 11 02 2011. [Online]. Available: <http://www.scholarpedia.org/article/Fisherfaces>. [Consultato il giorno March 2018].
- [9] K. Salton, «Face Recognition: Understanding LBPH Algorithm,» *Towards Data Science*, 10 November 2017. [Online]. Available: <https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b>. [Consultato il giorno February 2018].
- [10] G. Osman, M. S. Hitam e M. N. Ismail, «Enhanced Skin Colour Classifier Using RGB,» *International Journal on Soft Computing*, vol. 3, n. 4, 2012.
- [11] S. Kolkur, D. Kalbande, P. Shimpi, C. Bapat e J. Jatakia, «Human Skin Detection Using RGB, HSV and YCbCr Color Models,» in *International Conference on Communication and Signal*, 2017.
- [12] W. R. Tan, C. S. Chan, P. Yogarajah e J. Condell, «A Fusion Approach for Efficient,» *IEEE*

TRANSACTIONS ON INDUSTRIAL INFORMATICS, vol. 8, n. 1, 2012.

- [13] R. Bandara, «Bag-of-Features Descriptor on SIFT Features with OpenCV (BoF-SIFT),» Code Project, 30 August 2017. [Online]. Available: <https://www.codeproject.com/Articles/619039/Bag-of-Features-Descriptor-on-SIFT-Features-with-O>. [Consultato il giorno May 2018].
- [14] U. Sinha, «SIFT: Theory and Practice,» AI Shack, 2017. [Online]. Available: <http://www.aishack.in/tutorials/sift-scale-invariant-feature-transform-introduction/>. [Consultato il giorno June 2018].
- [15] P. J. Grother, M. L. Ngan, G. W. Quinn, «Face In Video Evaluation (FIVE) Face Recognition of Non-Cooperative Subjects,» National Institute of Standards and Technology, 2017.
- [16] E. Yiallourou, R. Demetriou, A. Lanitis, «On the Detection of Images Containing child-pornographic material,» IEEE International Conference on Telecommunications, Limassol, 2017.
- [17] K. Meethongjan, D. Mohamad, «A Summary of literature review : Face Recognition,» Postgraduate Annual Research Seminar 2007, Skudai, 2007.
- [18] S. Dutta, V.B. Baru, «Review of Facial Expression Recognition System and User Datasets,» *International Journal of Research in Engineering and Technology*, vol. 02, n. 12, pp. 641-645, 2013.
- [19] B. C. Ko, «A Brief Review of Facial Emotion Recognition Based,» *Sensors*, 2018.
- [20] T. Kanade, J.F. Cohn, Yingli Tian, «Comprehensive Database for Facial Expression Analysis,» in *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, Grenoble, 2000.
- [21] K. B. Shaika, P. Ganesana, V. Kalista, B. S. Sathisha, J. Merlin Mary Jenitha, «Comparative Study of Skin Color Detection and Segmentation in HSV and YCbCr Color Space,» in *3rd International Conference on Recent Trends in Computing*, 2015.
- [22] A. Elgammal, C. Muang, D. Hu, «Skin Detection - A Short Tutorial,» Springer, Berlin, 2009.
- [23] F. Saxen, A. Al-Hamad, «Color-Based Skin Segmentation: An Evaluation of the State of the Art,» in *2014 IEEE International Conference on Image Processing (ICIP)*, Paris, 2014.
- [24] K. Nallaperumal, S. Ravi, C. Nelson Kennedy Babu, R.K. Selvakumar, A. Lenin Fred, Seldev Christopher, S.S. Vinsley, «Skin Detection using Color Pixel Classification with Application to F

- ace Detection: A Comparative Study,» in *International Conference on Computational Intelligence and Multimedia Applications*, Sivakasi, India, 2007.
- [25] Z. Talukder, A. Basak, M. Shoyaibr, «Human Skin Detection,» *Global Journal of Computer Science and Technology*, vol. 13, n. 3, pp. 31-37, 2013.
- [26] M. R. Mahmoodi et al., «SDD: A skin detection dataset for training and assessment of human skin classifiers,» in *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, Tehran, Iran, 2015.
- [27] A. Rosebrock, «Skin Detection: A Step-by-Step Example using Python and OpenCV,» Pyimagesearch, 18 August 2014. [Online]. Available: <https://www.pyimagesearch.com/2014/08/18/skin-detection-step-step-example-using-python-opencv/>. [Consultato il giorno May 2018].
- [28] A. P. B. Lopes, S. E. F. de Avila A. N. A. Peixoto, R. S. Oliveira, A. de A. Araujo «A Bag Of Features Approach Based On HUE-SIFT Descriptor For Nude Detection,» in *17th European Signal Processing Conference (EUSIPCO 2009)*, Glasgow, Scotland, 2009.