Funded by the Horizon 2020 Framework Programme of the European Union
SocialTruth - Grant Agreement 825477

D3.3 SocialTruth Deep Learning Multimedia Verification

Dissemination Level:	**PU**
Nature of the Deliverable:	**R**
Date:	**30/09/2019**
Distribution:	**WP3**
Editors:	**LSBU**
Reviewers:	**QWANT, ESF**
Contributors:	**ALL**

Abstract:

The main aim of this deliverable is to specify the framework for the design and development of various image and video verification techniques using several approaches, involving advanced image processing as well as advanced visual tracking techniques across multiple video frames to assess the level of similarities within videos.

**** Dissemination Level:*** *PU= Public, RE= Restricted to a group specified by the Consortium, PP= Restricted to other program participants (including the Commission services), CO= Confidential, only for members of the Consortium (including the Commission services)*

***** Nature of the Deliverable:*** *P= Prototype, R= Report, S= Specification, T= Tool, O= Other*

Disclaimer

.

Revision History

Date	Rev.	Description	Partner
10/07/2019	0.1	First version of DDP	LSBU
15/07/2019	0.3	Initial deliverable structure (TOC)	LSBU
30/07/2019	0.4	Initial deliverable Draft	LSBU, UTP
2/09/2019	0.5	Aggregation of partners feedback	LSBU, UTP
11/09/2019	0.6	Updated version after periodic technical meeting (remotely)	ESF, QWANT, UTP, ICCS, LSBU
18/09/2019	0.8	Updated version after periodic technical meeting (remotely)	ESF, QWANT, UTP, ICCS, LSBU
24/09/2019	1.0	Updated version after periodic technical meeting. Extension to completed document structure (remotely)	LSBU
28/09/2019	2.0	Updated version, based on peer-review feedback	LSBU, UTP
29/09/2019	2.2	Further improvements with formatting	LSBU, ICCS
30/09/2019	2.3	Final version for submission	LSBU, UTP

List of Authors

Partner	Author
LSBU	Chathura Galkandage, Manik Gupta
UTP	Michał Choraś, Rafal Kozik
ICCS	Kostas Demestichas, George Koutalieris

Executive Summary

Deep learning based multimedia verification has started its journey recently and moving towards integrative models where more aspects of multimedia can be considered. We in this SocialTruth project investigate Image verification and Fake Video Analysis as the two main aspects of above. In terms of image verification research, there are three main identified forgery types are there. In literature, we find only deep learning based models, which can detect only one type of image forgery. Therefore, there is a real need to have a more generic solution, which can detect different types of image forgeries.

Here we propose individual modules of image forgery detection by improving existing deep learning models. On top of that, a new module to classify forgery types and some integrator to combine outcomes of individual forgery detections are proposed. The implementation of these researches are still underway and heading towards some remarkable results.

In terms of the fake video analysis, a good understanding of literature methods utilizing deep learning is very important as a starting point. Detection of forged frames, duplicate frames and deepfake videos are main research areas of this topic.

Looking at the big picture of the SocialTruth verification eco-system, the work described here will be used as independent provides some important inputs to other modules. The API with functionalities for different customer requirements is also addressed in this deliverable to ease the integration of this work in to the main stream of this project.

Table of Contents

Contents

1 Introduction (LSBU)

This document consists the outcome from the following task:

- Task 3.4: Image Verification & Video Fake Analysis – led by LSBU,

1.1 Motivation

Task 3.4 is responsible to define various image verification techniques using several different techniques like contextual image analysis, person re-identification with in fake images and publisher verification based on the historical social media posts. Further support to visual information will be acquired from multimedia meta data such as camera information, history of the file and encoding types. A deep learning based approach will be considered to achieve higher accuracy at the cost of finding substantial multimedia data to train neural models.

1.2 Intended audience

This deliverable is a report produced for all the members of the SocialTruth project. Specifically, the results of this report are addressed to the following audience:

- End-user partners, who will deploy elements of the SocialTruth platform and its particular components,
- The SocialTruth project researchers and developers, who will provide technical solutions,
- The platform integrators.

1.3 Scope

In general, the purpose of D3.3 document is to provide state of the art as well as the progress on the outputs of Task 3.4.

The D3.3 will be divided into three parts, focusing on:

- Image based analysis: Identification of important image parts in social images mostly vulnerable to forging is categorized in to a limited number.
- Video based analysis: Temporal dynamics needs to be a minimum to apply light-weight image processing techniques on less occurring time-specific forging.
- Meta verification: Standardized set of data is required to define to integrate with image and video based detection results.

1.4 Relation to other deliverables

This deliverable is linked with other deliverables produced within the SocialTruth project. D3.1 and D3.2 will be related in terms of meta-data analytics and content analysis required in D3.3.

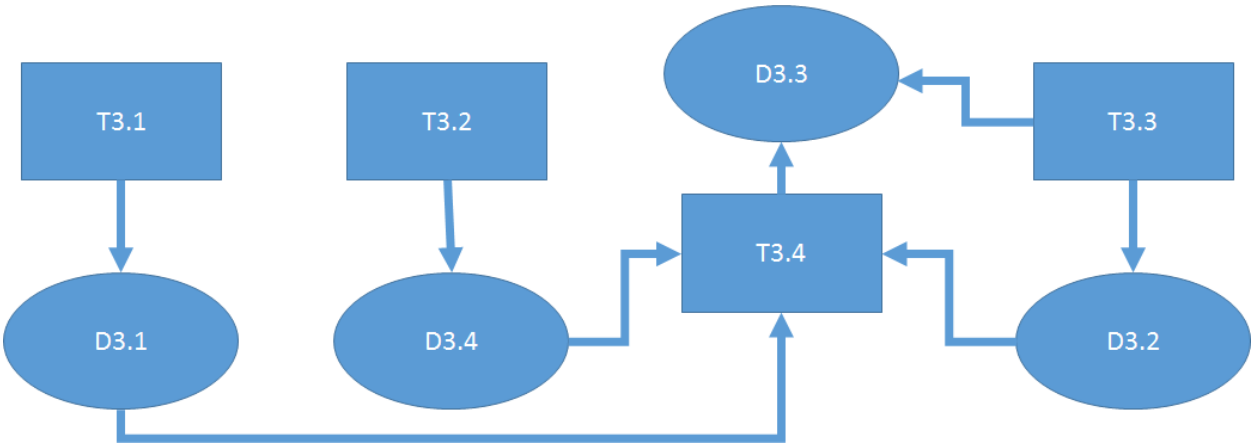

Fig. 1 Relation of D3.3 with other deliverables and project tasks

The D3.3 deliverable produces outcomes to the following deliverables:

- D3.4 SocialTruth Content Analysis and Verification Services – Release 1
- D3.5 SocialTruth Content Analysis and Verification Services – Release 2
- D5.1 Overall Evaluation Plan
- D5.2 SocialTruth Integrated Prototypes

2 Contextual Image Verification (LSBU)

With the ease of access to photo editing software, social networking has faced the challenge of verifying its images for forgeries. The type of image forgery/editing is very important to understand the underlying changes in an image. Therefore it is important to classify these tampering methods and kind of verification methods developed for each such image forgery type.

2.1 Literature survey (LSBU)

Mainly there are three different tampering types namely copy-move, cut-paste and erase-fill. Different names exist for these image forgeries such as splicing or removal for latter two types respectively. All these things count under image forgeries, which is the main branch of image manipulation where image steganography can also considered.

- Post-processing methods

Some early works of image tampering detection did not consider image post-processing after one of the three tampering operations. For instance, splicing was defined as the simple cut-paste operation of image regions from one image onto the same or another image without performing post-processing [1]. However, practical tampering often involves post-processing operations to smooth the boundaries of tampered regions, in order to make the final artefact less visually susceptible.

There are two kinds of post-processing operations. One is **active post-processing** for improving the tampering effect, e.g., image blurring, image resampling, brightness change and contrast adjustments.

The other one is **passive post-processing** that may be unintentionally introduced to tampered images during data transmission, e.g., JPEG compression, noise adding and color reduction [2]. Nowadays, when we refer to image tampering, we imply all the tampering processes with or without post-processing.

- Deep learning based approaches

A summary of existing deep learning models is given below to understand the justification of finding our base method as the most robust method from literature.

Table 1. Existing deep learning models for forged image detection

Method	Main summary points
Image Manipulation Detection using Deep Siamese Convolutional Neural Network [3]	• Employs a **deep siamese CNN**, which has twin CNNs accepting two image patches as the input and classifies the patch pair as either identically processed (IP) or differently processed (DP). • Different image editing operations: Gaussian blurring, median filtering, resampling, corrupting with the additive white Gaussian noise (AWGN), gamma correction.
Learning Rich features for	• **Faster R-CNN** two stream network

Image Manipulation detection [4]	• Features from RGB channel to capture clues like visual inconsistencies at tampered boundaries and contrast effect between tampered and authentic regions • Second stream analyzes the local noise features in an image – SRM filter kernels to produce noise features
Deep Learning approach to image region forgery detection [5]	• Apply 3 level 2D Daubechies Wavelet Decomposition to each YCrCb component of the patches • Std dev, mean, sum for each approximation, horizontal, vertical and diagonal coefficients to obtain 90 features • Daubechies Orthogonal wavelets D2-D5 to obtain 450 features • Stacked Autoencoder for complex feature learning • Context learning for tampered regions
BusterNet: Detecting Copy-Move Image Forgery with Source/Target Localization [6]	• End-to-end trainable, deep neural network solution • First model to localize source/target regions • Features a two-branch architecture followed by a fusion model • Also provides synthesizing large-scale CMFD samples using out-of-domain datasets

- Introduction to databases

  Above existing methods utilizes following databases. Our proposed method is also tested using selected databases from here.

Table 2. Summary of databases used in image forgery detection models

Database	Tampering type	# Authentic/#Tampered	Mask	Post-processing
Columbia2006 [7]	Cut-paste	183/180	Yes	No
CASIAv2.02009[8]	Cut-paste copy-move	7491/5123	No	Yes
MICC-F60002013[9]	Copy-move	440/160	Yes	Yes
IMD 2012 [10]	Copy-move	48/48	Yes	Optional
CoMoFoD2013 [2]	Copy-move	5200/5200	Yes	Yes
WildWeb2015 [11]	Cut-paste copy-move Erase-fill	0/10646	Yes	Yes
COVERAGE 2016 [12]	Copy-move	100/100	Yes	No

2.2 Research challenges

Some of the research challenges involved in this task are mentioned as follows:

- Integration of existing solutions to create a single solution for any forgery type

A summary regarding the understanding of detection clues individually and how they map into different forgery type is mentioned in the table below. Understanding detection clues individually and map them to different forgery type is performed here.

Table 3. Detection clues to identify different forgery types

Detection clue	Copy-move	Cut-paste	Erase-fill	Localization
Region duplication	Yes	No	Exemplar-based	Yes
Edge anomaly	Yes	Yes	No	No
Sharp edges	Yes	Yes	No	No
Blurred edges	Yes	Yes	No	Yes
Region anomaly				
JPEG DQ inconsistency	No	Yes	Yes	Yes
Lighting inconsistency	No	Yes	No	Yes
Camera trace inconsistency	No	Yes	No	Yes
Blurred region	No	Yes	Diffusion-based	Yes
Median filtering inconsistency	No	Yes	No	Yes
Re-sampling inconsistency	Yes	Yes	No	Yes

- Inclusion of computer vision techniques to optimize existing deep learning methods

Optimization of above detection methods using computer vision related research is another avenue in this project. Better algorithms for each detection clue is considered here. For example, looking at region duplication only possible areas of duplication can be understood using visual attention models. There will be different cases to add for other detection clues. As for now we are only looking at how visual attention models going to help the course of the project.

2.3 Deep learning approach overview

In this section we describe the approach that we have adopted for our solution for deep learning based multimedia verification based on the Busternet solution.

BusterNet is a pure, end-to-end trainable, deep neural network solution. It features a two-branch architecture followed by a fusion module. The two branches localize potential manipulation regions via visual artefacts and copy-move regions via visual similarities, respectively.

It uses a DNN pipeline that is (i) end-to-end trainable, such that it does not include manually tuned parameters and/or decision rules and (ii) capable of producing distinct source and target manipulation masks (which could be used for forensic analysis).

To achieve the above goals, a valid DNN solution should attain two feature properties simultaneously, (i) source and target features are dissimilar enough to distinguish source from target, and (ii) they are also more similar than those in pristine regions. Of course, one can train a naive DNN, while hoping it could attain these properties magically. However, a better idea is to explicitly consider these properties, and we therefore adopt BusterNet, a two-branch DNN architecture as shown in the Figure 2.

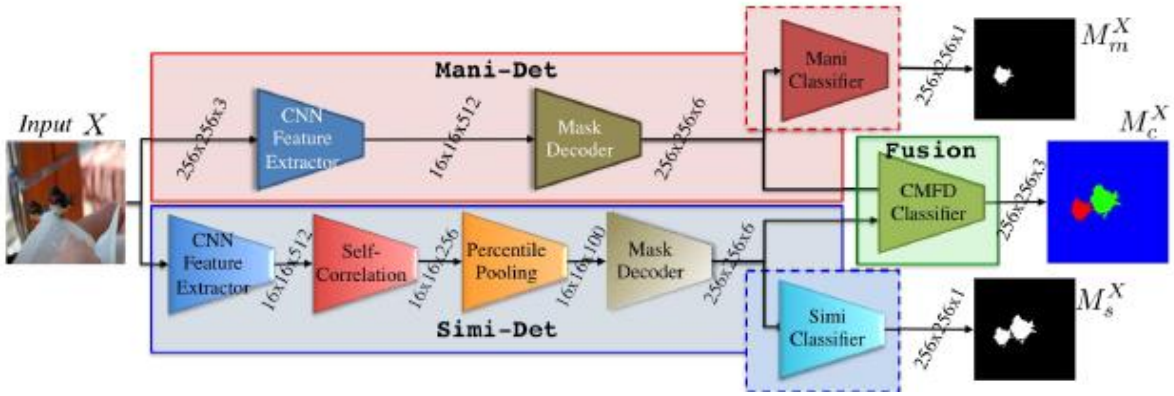

Fig. 2 Overview of the proposed two-branch DNN-based CMFD solution

Dashed blocks are only activated during branch training. Output mask of the main task, i.e.M_c^X , is color coded to represent pixel classes, namely pristine (blue), source copy (green), and target copy (red). Output masks of auxiliary tasks, i.e.M_m^X and M_s^X , are binary where white pixels indicates manipulated/similar pixels of interests, respectively.

Specifically, BusterNet designs Mani-Det branch to detect manipulated regions such that its feature is good for property (i), while Simi-Det branch to detect cloned regions such that its feature attains property (ii), and finally use both features in Fusion to predict pixel-level copy-move masks differentiating pristine, source copy, and target copy classes. To ensure these two branches achieve the desired functionality, we define each branch an auxiliary task, as indicated by the dashed blocks in Fig. 2. More precisely, Mani-Det's and Simi-Det's tasks are to predict a binary manipulation mask M_m^X and a binary copy-move mask M_s^X , respectively, and both binary masks can be derived from the 3-class mask M_c^X.

- Justification on the selection of BusterNet

This is the first copy-move forgery detection (CMFD) algorithm with discernibility to localize source/target regions. It also proposes simple schemes for synthesizing large-scale CMFD samples using out-of-domain datasets, and stage-wise strategies for effective BusterNet training. Its extensive studies demonstrate that BusterNet outperforms state-of-the-art copy-move detection algorithms by a large margin.

2.3.1 Detection of manipulated regions

- Convolutional feature extracting

The manipulation detection branch (i.e. Mani-Det as shown by red shaded regions in Fig. 2) can be thought of as a special segmentation network whose aim is to segment manipulated regions. More precisely, it takes input image X, extracts features using CNN Feature Extractor, up-samples the feature maps to the original image size using Mask Decoder, and applies Binary Classifier to fulfill the auxiliary task, i. e. producing a manipulation mask Mm^X.

Any convolutional neural network (CNN) can serve as CNN Feature Extractor. Here, we use the first four blocks of the VGG16 architecture [13] for its simplicity. The resulting CNN feature fm^X is of size 16×16×512, whose resolution is much lower than that is required by the manipulation mask.

- Mask decoding

We need to decode this feature, and apply deconvolution [14] to restore the original resolution via the Mask Decoder as shown Fig. 3, which applies BN-Inception and BilinearUpPool2D [15] in an alternating way and eventually produces a tensor dXm of shape 256×256×6. To be clear, 16 times of the spatial dimension increase is due to the 4 times of BilinearUpPool2D (i.e. 2^4=16), and the output filter dimension 6 is because of the last BN-Inception(2@[5,7,11]), which concatenates 3 Conv2D responses, each with 2 output filters but uses kernel sizes at (5,5), and (7,7) and (11,11), respectively (i.e. 3×2=6). Finally, we predict pixel-level manipulation mask MX m via Binary Classifier, which is as simple as a single Conv2D layer with 1 filters of kernel size (3,3) followed by the sigmoid activation.

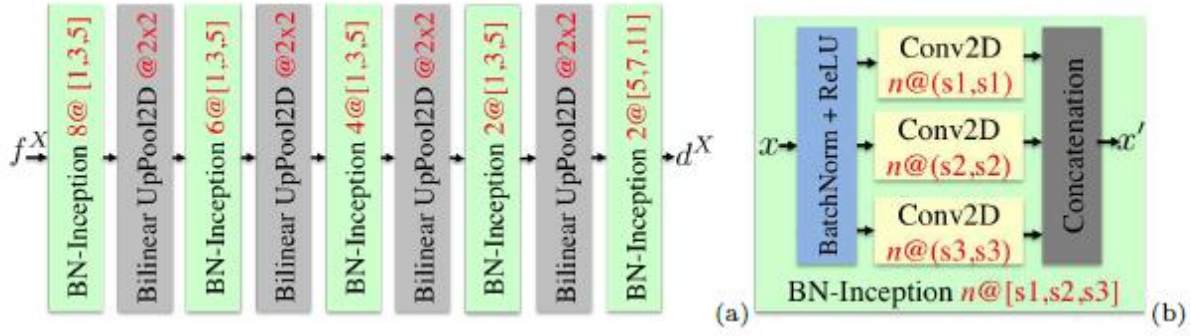

Fig. 3 Inception-based mask Deconvolution module. (a) Mask deconvolution network and (b) parametric BN-inception module, where s1, s2 and s3 indicates the kernel sizes used in three Conv2D layers, respectively, and n stands for the number of filters.

2.3.2 Detect cloned regions

- Self-correlation

The similarity detection branch (i.e. Simi-Det as shown by blue shaded regions in Fig. 2) takes an input image X, extracts features using CNN Feature Extractor, computes feature similarity via Self-Correlation module, collects useful statistics via Percentile Pooling, up-samples feature maps to the original image size using Mask Decoder, and applies Binary Classifier to fulfil the auxiliary task, i.e. producing a copy-

move mask Mm^X at the same resolution of X. It is worthy to stress that modules shared in both branches, e.g. CNN Feature Extractor, only share the network architecture, but not weights.

Like Mani-Det branch, Simi-Det branch starts with feature representation via CNN Feature Extractor. It again produces a feature tensor fs^X of size 16×16×512, which can be also viewed as 16×16 patch-like features, i.e. $fs^X = \{fs^X[ir, ic]\}_{ir,ic \in [0, \cdots ,15]}$, and each with 512 dimensions. Since our goal is to recover the potential copy-move regions, we have to mine useful information to decide what matched patch-like features are. To do so, we first compute all-to-all feature similarity score using Self-Correlation, and collect meaningful statistics to identify matched patches via Percentile Pooling.

- Percentile pooling

Another advantage of the above standardization is dimension reduction, because only a small portion of all scores is kept. Once Percentile Pooling is done, we use Mask Decoder to gradually up sample feature PX to the original image size as ds^X , and Binary Classifier to produce a copy-move mask Ms^X to fulfill the auxiliary task. Again, both Mask Decoder and Binary Classifier only have the same architecture as those in Mani-Det, but with distinctive weights.

2.3.3 Feature fusion

As illustrated in Fig. 2, Fusion module takes inputs of the Mask Decoder features from both branches, namely dm^X and ds^X , and jointly considers these two branches and make the final CMFD prediction. More precisely, we (i) concatenate feature dm^X and ds^X , (ii) fuse feature using the BN-Inception with parameter set 3@[1,3,5] (see Fig. 3-(b)), and (iii) predict the three-class CMFD mask using a Conv2D with one filter of kernel size 3×3 followed by the softmax activation.

2.4 Results

Current implementation of BusterNet on CASIA and CoMoFoD databases are illustrated in this section. A summary of the model is presented in Fig. 4 while an example of the model performance on localizing duplicate objects is shown below in Fig.5. Some performance values of the models are presented for given databases in Fig. 6.

```
_________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
=================================================================================================
image_in (InputLayer)        (None, None, None, 3 0
_________________________________________________________________________________________________
preprocess (Preprocess)      (None, 256, 256, 3)  0           image_in[0][0]
_________________________________________________________________________________________________
simiFeatex (Model)           (None, 256, 256, 6)  7735568     preprocess[0][0]
_________________________________________________________________________________________________
maniFeatex (Model)           (None, 256, 256, 6)  7789694     preprocess[0][0]
_________________________________________________________________________________________________
merge (Concatenate)          (None, 256, 256, 12) 0           simiFeatex[1][0]
                                                              maniFeatex[1][0]
_________________________________________________________________________________________________
fusion_c0 (Conv2D)           (None, 256, 256, 3)  39          merge[0][0]
_________________________________________________________________________________________________
fusion_c1 (Conv2D)           (None, 256, 256, 3)  327         merge[0][0]
_________________________________________________________________________________________________
fusion_c2 (Conv2D)           (None, 256, 256, 3)  903         merge[0][0]
_________________________________________________________________________________________________
fusion_merge (Concatenate)   (None, 256, 256, 9)  0           fusion_c0[0][0]
                                                              fusion_c1[0][0]
                                                              fusion_c2[0][0]
_________________________________________________________________________________________________
fusion_bn (BatchNormalization) (None, 256, 256, 9)  36          fusion_merge[0][0]
_________________________________________________________________________________________________
fusion_re (Activation)       (None, 256, 256, 9)  0           fusion_bn[0][0]
_________________________________________________________________________________________________
pred_mask (Conv2D)           (None, 256, 256, 3)  246         fusion_re[0][0]
_________________________________________________________________________________________________
restore (ResizeBack)         (None, None, None, 3 0           pred_mask[0][0]
                                                              image_in[0][0]
=================================================================================================
Total params: 15,526,813
Trainable params: 15,526,007
Non-trainable params: 806
_________________________________________________________________________________________________
None
```

Fig. 4 Model summary

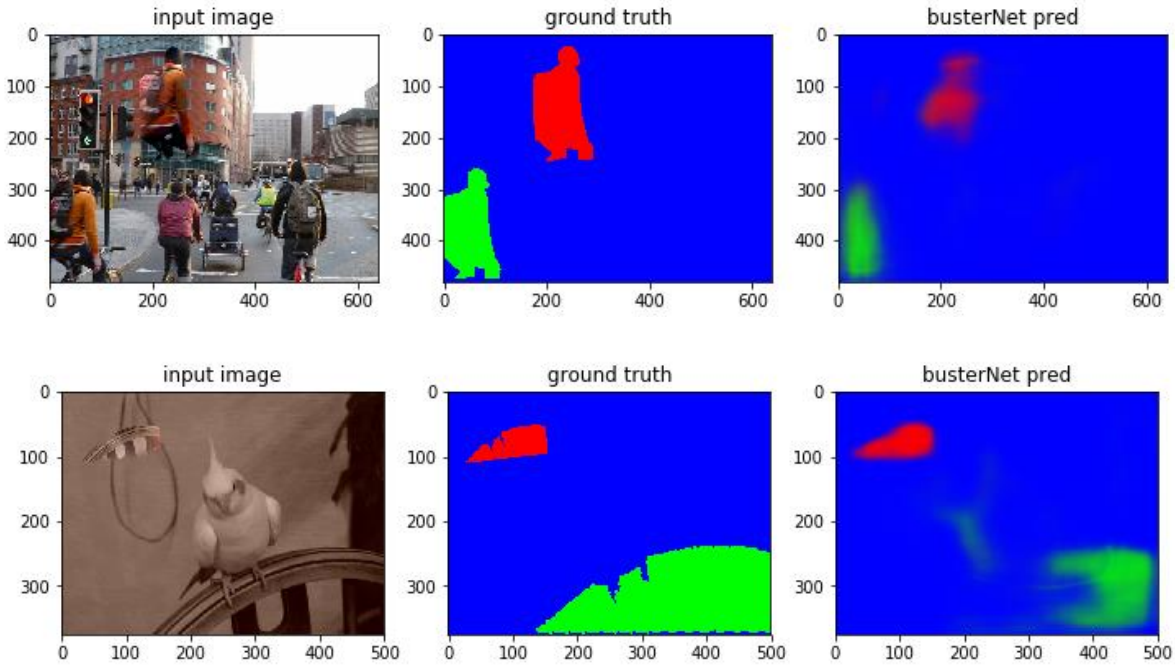

Fig. 5 Comparison of localization performance against ground truth data for samples

```
INFO: BusterNet Performance on CASIA-CMFD Dataset using Pixel-Level Evaluation Protocal-B
--------------------------------------------------------------------------------------------------------
INFO: Precision = 0.557
INFO:     Recll = 0.438
INFO:        F1 = 0.456


INFO: BusterNet's Discernibility Performance Analysis
--------------------------------------------------------------------------------------------------------
Total        = 1313
Miss         = 542
OptOut       = 580
OptIn        = 191
Correct      = 145
--------------------------------------------------------------------------------------------------------
Overall-Acc. = 0.110
OptIn-Acc.   = 0.759
```

Fig. 6 BusterNet performance on CASIA database using Pixel-Level Evaluation and Discernibility analysis

2.5 Visual Attention models

- Different complexity levels

Based on the verification requirements we can either use a simple filtering technique such as saliency maps to simplify our BusterNet solution or introduce another neural network to build up a dynamic internal representation of the image.

- Saliency maps – Filters to run on neural networks

This is much simpler approach which simplifies the inputs of chosen deep learning method of BusterNet. Instead of feeding the whole image for similarity checks, here we propose to feed block based matching with areas with high saliency importance to check first. In this way similarity checking would start hitting the clone regions much faster.

- Recurrent models of visual attention

This is a recurrent neural network that processes inputs sequentially, attending to different locations within the image one at a time, and incrementally combining information from these fixations to build up a dynamic internal representation of the image.

2.6 Image verification based on texture and meta-data analysis

Our proposed method[1] is based on the image assessment. The assumption is that if the image is corrupted, the whole news may be fake. In the algorithm we have three factors evaluated:

- ELA analysis,
- copycat searching,
- meta data of file analysis.

[1] Choraś M., Giełczyk A., Demestichas K., Puchalski D., Kozik R. (2018) Pattern Recognition Solutions for Fake News Detection. In: Saeed K., Homenda W. (eds) Computer Information Systems and Industrial Management. CISIM 2018. Lecture Notes in Computer Science, vol 11127. Springer, Cham

Fig. 7 Samples from the CASIA database (first row contains original photos, second row contains modified photos, the lowest row contains images modified with the copy-paste method

In the project we used the image manipulation database which is available online[2]. It consists of 800 unmodified images (original) and 921 images with additional elements added. 25 images were classified as modified by cloning. Some samples from the database are presented in **Σφάλμα! Το αρχείο π ροέλευσης της αναφοράς δεν βρέθηκε.**.

To detect pasted elements, the analyzed image is saved on the disc and read again. Then, the absolute value of difference between corresponding pixel is calculated and multiplied by the scale factor.

In order to detect cloning, the image is divided into overlapping blocks. Then, to extract features from blocks, the SURF algorithm was performed and FLANN algorithm for matching.

Meta data analysis may be performed using various libraries. They can give the information about modification done by any image processing tool. However, not all modified images by means of Photoshop are fake images.

[2] https://www.kaggle.com/sophatvathana/casia-dataset

That is why, the final decision is based on the result of logic function F expressed with equation below, where x – ELA decision, y – copycat decision, z – meta data decision and x,y,z ϵ {true,false}. Values of F, x, y and z are false, when the image is assessed as modified and true otherwise.

$$F(x, y, z) = x \cdot y + y \cdot z + z \cdot x$$

For the research assessment, the accuracy and FAR, FRR measures were used. Accuracy is expressed with Eq.2., where TP – modified images assessed as modified and FP – not modified images assessed as unmodified, N – number of samples.

FAR (False Acceptance Rate) gives the information about the number of defrauded images classified as unmodified and is expressed with Eq.3., where FP – number of incorrectly classified unmodified images and NP – total number of unmodified images.

FRR (False Rejection Rate) tells about the number of unmodified images classified as defrauded and is expressed with Eq. 4., where FN – number of incorrectly classified modified images and NN – total number of modified images.

When FRR and FAR are equal, the other measure may be introduced, namely EER (Equal Error Rate), which is equal to FAR=FRR.

$$Acc = \frac{TP + TN}{N} \cdot 100\% \quad (2)$$

$$FAR = \frac{FP}{NP} \quad (3)$$

$$FRR = \frac{FN}{NN} \quad (4)$$

Table shows the accuracy of the SURF-based part of the system. The accuracy depends on the Hessian threshold value, which was set experimentally.

Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε. presents the EER measure of the ELA part o f the system. The ELA analysis was performed for different scale parameters (20 and 40) and two quality parameters (75 and 100). By combining the most promising results it was possible to obtain the 64% accuracy of image forgery detection.

Table 4 Accuracy of SURF-based part of the system

Hessian threshold	Accuracy
100	76%
400	74%
1500	64%

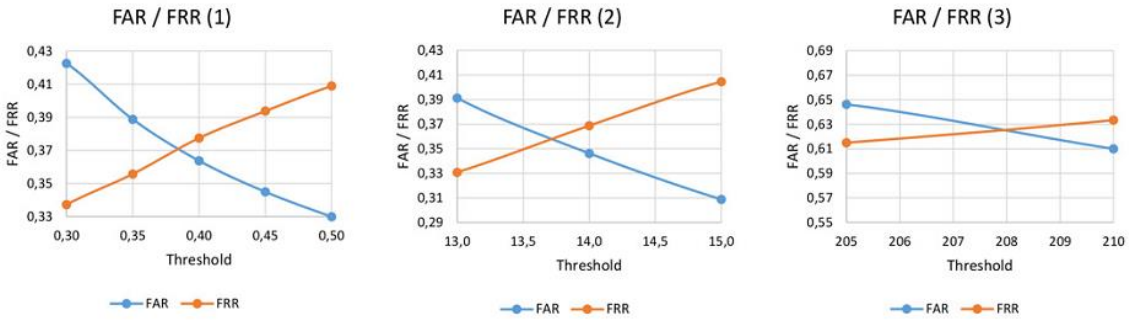

Fig. 8 FAR/FRR results of ELA part of the system for various scale and quality parameters

2.7 Forgery classifier

A more detailed understanding of correlation between visual attention and type of forgery is considered here. A basic computer vision check on the type of forgery like to be available for a given content type. The development of the algorithm for this module is under investigation.

2.7.1 Detect erase-fill regions

Identifying a deep learning network model for erase-fill detection is under investigation.

2.7.2 Detect foreign camera traces

Identifying image processing methods for camera tracing to detect cut-paste forgery is under investigation.

2.7.3 Building a classifier

Source modelling using machine learning separate forgery types is considered here as per Table 3. In order to improve the accuracy of the classifier Iterative binary classification is used with simple algorithms such as support vector machines.

2.8 Integrated solution for all forgery types

This module is currently under investigation and estimated to utilize the outcomes of the forgery classifier to merge individual outcomes of different types of forgery detectors.

2.9 The operational ecosystem

2.9.1 Distributed image verification system

Our system of image verification stands as a main component of the multimedia verification system. The API provided by the proposed solution facilitates functionalities such as copy-move check, cut-paste check, erase-fill check, post-processing identifier and other special types of services. Spliced face identification, duplicate object detection, photo authentications are a few such functions to name.

2.9.2 The technical architecture

The modular architecture of the eco-system is illustrated below in Fig. 7. The outcome of the post-processing correction module will be fed in to the forgery classifier which forwards images to parallel modules of copy-move, cut-paste and erase-fill detectors with an expected weight for each forgery type. After the individual verification results have achieved, the integrator would produce a normalized verification result based on the probabilities of forgery classifier. Finally the eco-system would look at addressing special functionalities of the API.

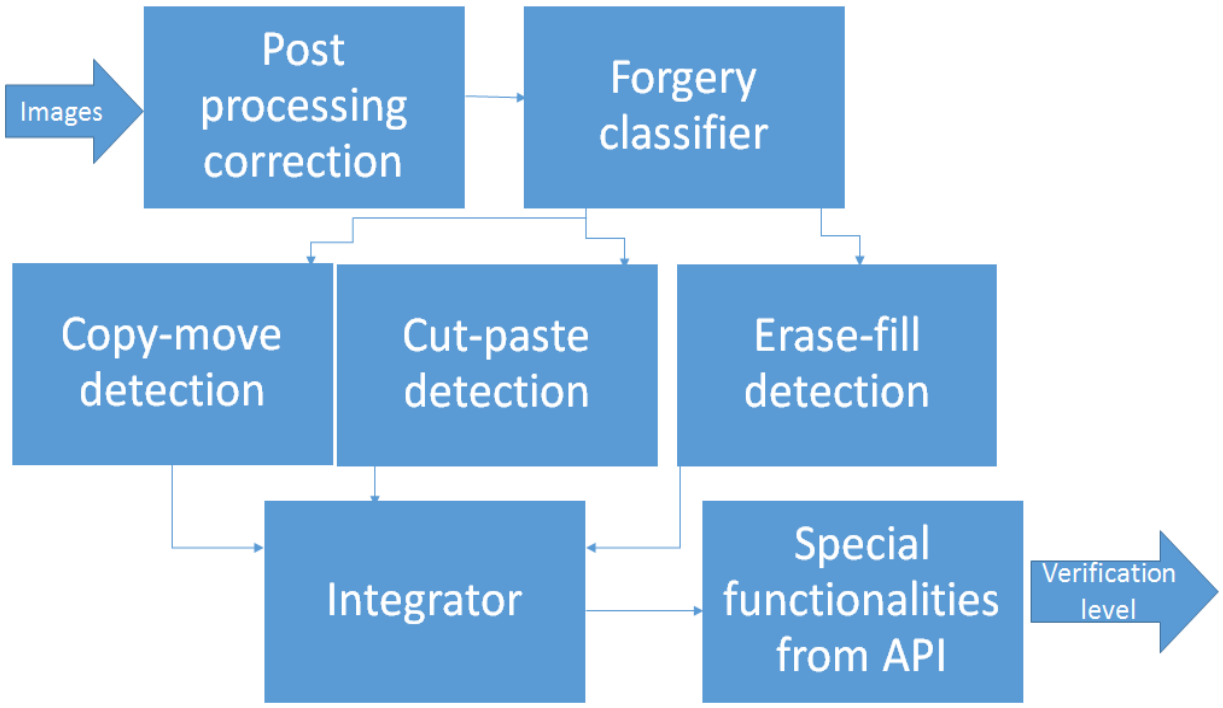

Fig. 9 Modular architecture of the image verification system

3 Fake Video Analysis

Investigation of literature is underway for this area of the research. Some important approaches are described in this chapter.

3.1 Detection of forge frame duplications

A passive-blind scheme for detection of frame duplication forgery in videos is considered here. This will be a multi-stage implementation where a coarse-to-fine approach is employed as found in [16].

1. Candidate segment selection: To screen and select duplicated candidates in the temporal domain, the histogram difference of two adjacent frames in the RGB color space is adopted as a feature.
2. Spatial similarity measurement: To evaluate the similarity of two images, we require a block-based algorithm to measure the spatial correlation between the candidate segment and the corresponding frame in the query template.
3. Frame duplication classification: Based on the results of spatial and temporal analysis, we construct a classifier to detect duplicated clips.
4. Post-processing: In addition, to deal with the partial detection problem, require to develop a post-processing technique that examines and merges two adjacent detected candidates into a complete duplicated video clip.

We expect our system will accurately detect and localize duplicated clips in different kinds of videos.

3.2 Detect forged frames using pattern noise

The detection of a forged frame in a video can be determined by comparing the correlation between the noise within the frame itself and the reference pattern noise with an empirical threshold. This is because the camera pattern noise is a unique stochastic high frequency characteristic of imaging sensors. The reference pattern is created for the identification of the camera is used for the authentication of the video. This pattern is created from the video sequence as time progresses, with a technique applied frame per frame, by averaging the noise extracted from each frame. By using this method we will be able to identify if all the scenes of a video sequence have been taken with the same camera and if the number and/or the content of the frames of the video have been modified. Some of the key steps involved are as follows:

1. The first part of the creation of the reference pattern is done on the first 50 frames of the video
2. Initially we extract the high frequency noise from the first image using a de-noising filter
3. In the same manner, the noises of the second frame and then of the successive frames are added and averaged to construct the reference pattern.

3.3 Temporal-aware pipeline to automatically detect deepfake videos

In recent months machine learning based free software tool has made it easy to create believable face swaps in videos that leaves few traces of manipulation, what are known as "deepfake" videos.

A temporal-aware pipeline to automatically detect deepfake videos will be considered here. This system uses a convolutional neural network (CNN) to extract frame-level features. These features are then used to train a recurrent neural network (RNN) that learns to classify if a video has been subject to manipulation or not.

This is an ongoing work and an update on the implementation and results will be reported as the project progresses.

4 External verification services

We are aware of some news verification services existing as free online tools, mainly used as a browser plugins or extensions. Among image analysis approach, in which content is verified against modification of images or using them in manipulated context (e.g. images from different locations or older images than described event), some other approaches are used, namely:

- metadata analysis - if the author provided the metadata or tag, those may be also used in order to classify the specific piece of information as true or fake
- journalist analysis - in this kind of approach the journalist former articles are analyzed, if there were any fake information before, the websites of trusted news agencies are less likely to produce fake online information in comparison to the newly created websites of private blogs
- text analysis - the news text may be compared to the news released on approximate time

We identified and analyzed several tools for automated fake news detection based on the fake news detection accuracy[3]. Analyzed tools use different approaches for fake news detection, such as image, author and textual analysis, however in most of cases they work as "black-box" – algorithms or models applied are not revealed. A short overview of tools is provided in Table 5.

Table 5: Selected external verification services

No	Tool	URL	Description
1.	**SurfSafe**	https://www.getsurfsafe.com/	The SurfSafe tool is a free browser extension released by RoBhat labs in August 2018 available online. The main idea of this tool is to compare the images from news to a database of images. The images stored in the database are culled from both trusted and fact-checking sites. If the image was modified or used in fake context, the whole news piece is considered to be fake. Moreover, the text analysis is performed - the text from the news is compared to the text found within the image on another site. The SurfSafe user may adjust the set of trusted websites.
2.	**Fake News Detector AI**	http://www.fakenewsai.com/	The Fake News Detector AI is a tool available online. Unfortunately, there

[3] Giełczyk A., Wawrzyniak R., Choraś M. (2019) Evaluation of the Existing Tools for Fake News Detection. In: Saeed K., Chaki R., Janev V. (eds) Computer Information Systems and Industrial Management. CISIM 2019. Lecture Notes in Computer Science, vol 11703. Springer, Cham

			is no information about the algorithms involved in the detection procedure, other than the short message 'use a neural network'. The online interface provides the information 'true', 'false' or 'unknown error' for each link that the user wants to verify.
3.	**TrustedNews**	https://trusted-news.com/	The next tool involved in the research is called TrustedNews and may be found online. It was released by the MetaCert organization in 2017. This tool can provide a wider set of results - trustworthy, untrustworthy, satire, biased, malicious, clickbait, generated and unknown. Trusted News is powered by the MetaCert Protocol, and it is claimed to use 'independent, politically objective data sources to measure the truthfulness of news content'.
4.	**Fake News Detector**	https://fakenewsdetector.org/en	The Fake New Detector is available online. It involves the feedback provided by the other users of the tool. It can give one of the following answers: legitimate (real), fake news, clickbait or biased. What is very important, this is an open source project and its repositories are available on the Github platform.
5.	**Fake News Guard**	http://fakenewsguard.com/index.html	This tool is a passive one working as a browser extension. It verifies any page visited by the user and any link displayed in Facebook. However, it is difficult to evaluate the way this tool works. Authors only claim that it combines the linguistic approach, network analysis and artificial intelligence. Apart of that, each user can report the source, if it is suspected of being fake.

6.	**Decodex**	https://www.lemonde.fr/verification/	Decodex is an online tool released in France. It labels the pieces of information as 'info', 'satire' and 'no information', which may alert the user about potential fake news. Apart from the tool, the detailed user guide is available on the website. The authors encourage users to verify the information before sharing it, to verify the source of the news (and use only the trustworthy sources) and to verify the image used in the specific context.

For the research purposes we have gathered a set of 20 websites providing news. Most of them (12) contain text written in the Polish language, while the rest of the news is provided in English. All the elements were manually classified and labelled as real, fake, fake clickbait or satire.

All the online tools introduced above were used during the research. We have installed them as browser extensions or have used the dedicated online interface. While using the extensions, it was necessary to visit the investigated websites. The web-based interfaces on the other hand allowed for simply pasting the web address before coming up with the evaluation. The classification performed by each tool is presented in Table 6, where the listing number of the website, manual classification and the tools' classification are given.

Table 6: News classification performed manually (Classification) and by various tools: 1 - SurfSafe, 2 - Fake News Detector AI, 3 - Trusted News, 4 - Fake News Detector, 5- Fake News Guard, 6 - Decodex

No.	Classification	1.	2.	3.	4.	5.	6.
1	fake		fake				
2	real		real				
3	fake	fake	fake				
4	fake		fake		fake		
5	real	real	real		real		
6	satire		real	satire	satire	satire	satire
7	fake	fake	fake			fake	
8	fake		fake				
9	clickbait	fake	fake		clickbait		
10	clickbait	fake	fake		fake	clickbait	
11	clickbait	fake	fake		clickbait		
12	satire		real	satire	clickbait	satire	satire
13	clickbait		fake		clickbait		
14	real	fake	real				

15	real	fake	real	real	real		real
16	clickbait		real		clickbait		
17	fake				fake		
18	satire		real	satire			
19	satire		fake	satire		satire	
20	clickbait		fake			fake	

Then the accuracy of the detection was estimated. It is expressed in equation below, where WL - well classified samples and N - total number of samples (here N = 20). We assumed that 'clickbait' is well classified when is labeled as 'clickbait' or 'fake'. The 'satire' is well classified when is labeled as 'satire' of 'fake'. 'Fake' and 'real' pieces of information are classified well only when the result is 'fake' or 'real', respectively.

$$Acc = \frac{WL}{N} \cdot 100\%$$

The detailed results of the accuracy are presented in Table 3. The table illustrates that the highest accuracy was obtained using the Fake News Detector AI (75%), while the poorest using Decodex (15%). The provided data demonstrates that the outperforming tool was very successful in detecting the real news (100%), promising results were achieved for fake news and clickbait, but far weaker for satire.

For detecting the satire the TrustedNews tool achieved the best results on our dataset - it has classified all satire samples correctly. However, for the other categories it was significantly less successful.

The average value and standard deviation are also presented in Table 7. All methods give the average accuracy close to 39%. Nevertheless, the standard deviation values are very high, because the obtained accuracy results are very different for each tool and each category.

Table 7: Accuracy for different tools: 1 - SurfSafe, 2 - Fake News Detector AI, 3 - Trusted News, 4 - Fake News Detector, 5 - Fake News Guard, 6 - Decodex for all samples and 4 separate categories: 'fake', 'real', 'satiric' and 'clickbait' with average and standard deviation

Category	1.	2.	3.	4.	5.	6.	AVG	STD DEV
All samples	40%	75%	25%	55%	25%	15%	39%	22%
'Fake'	33%	83%	0%	50%	0%	0%	28%	34%
'Real'	75%	100%	25%	50%	0%	25%	46%	37%
'Satire'	0%	25%	100%	25%	75%	50%	46%	37%
'Clickbait'	50%	83%	0%	83%	33%	0%	42%	39%

The dataset contains sources written in the Polish and English languages. In Table 8 the accuracy of the detection is presented once again. It illustrates the dependency between the language of the content and the obtained accuracy. It is worth focusing on two tools: TrustedNews and Fake News Guard – they

provide over 60% for English content but give 0% for Polish sources. Thus, it is possible that these two tools do not contain non-English samples in the training set or database.

Table 8: Accuracy for different tools: 1 - SurfSafe, 2 - Fake News Detector AI, 3 -Trusted News, 4 - Fake News Detector, 5 - Fake News Guard, 6 - Decodex for both languages and each language separately

Language	1.	2.	3.	4.	5.	6.
Both	40%	75%	25%	55%	25%	15%
Polish	42%	83%	0%	58%	0%	0%
English	38%	63%	63%	50%	63%	38%

5 Conclusions

As per the current progress, image verification system is nearly completed with copy-move detection module successfully implemented. Further cut-paste and erase-fill modules are due to be completed in few months. Upon completing individual detection modules, forgery classifier and integration of verification results will be implemented. The development of algorithms for above work is under investigation. Fake video analysis is at the literature survey stage where we are going to decide on which methods can be integrated in to our proposed image verification eco-system.

References

[1] W. Wang, J. Dong, T. Tan, A survey of passive image tampering detection, IWDW, vol. 9, Springer, 2009, pp. 308–322.

[2] D. Tralic, I. Zupancic, S. Grgic, M. Grgic, CoMoFoD – new database for copymove forgery detection, in: Proceedings of ELMAR, 2013.

[3] Mazumdar A, Singh J, Tomar YS, Bora PK. Universal Image Manipulation Detection using Deep Siamese Convolutional Neural Network. arXiv preprint arXiv:1808.06323. 2018 Aug 20.

[4] Zhou P, Han X, Morariu VI, Davis LS. Learning rich features for image manipulation detection. InProceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2018 (pp. 1053-1061).

[5] Zhang Y, Goh J, Win LL, Thing VL. Image Region Forgery Detection: A Deep Learning Approach. InSG-CRC 2016 Jan 15 (pp. 1-11).

[6] Wu Y, Abd-Almageed W, Natarajan P. BusterNet: Detecting copy-move image forgery with source/target localization. InProceedings of the European Conference on Computer Vision (ECCV) 2018 (pp. 168-184).

[7] Y.F. Hsu, S.F. Chang, Detecting image splicing using geometry invariants and camera characteristics consistency, in: Proceedings of ICME, 2006.

[8] J. Dong, W. Wang, T. Tan, CASIA image tampering detection evaluation database, in: Proceedings of ChinaSIP, IEEE, 2013, pp. 422–426.

[9] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, G. Serra, A SIFT-based forensic method for copy–move attack detection and transformation recovery, IEEE Trans. Inf. Forensics Secur. 6 (3) (2011) 1099–1110.

[10] V. Christlein, C. Riess, J. Jordan, C. Riess, E. Angelopoulou, An evaluation of popular copy-move forgery detection approaches, IEEE Trans. Inform. Forensics Secur. 7 (6) (2012) 1841–1854.

[11] M. Zampoglou, S. Papadopoulos, Y. Kompatsiaris, Detecting image splicing in the wild (web), in: Proceedings of ICMEW, IEEE, 2015, pp. 1–6.

[12] B. Wen, Y. Zhu, et al., COVERAGE: a novel database for copy-move

[13] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)

[14] Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1520–1528 (2015)

[15] Wojna, Z., Ferrari, V., Guadarrama, S., Silberman, N., Chen, L.C., Fathi, A., Uijlings, J.: The devil is in the decoder (2017)

[16] Lin GS, Chang JF. Detection of frame duplication forgery in videos based on spatial and temporal analysis. International Journal of Pattern Recognition and Artificial Intelligence. 2012 Nov 9;26(07):1250017.

[17] N. Mondaini, R. Caldelli, A. Piva, M. Barni, and V. Cappellini "Detection of malevolent changes in digital video for forensic applications", Proc. SPIE 6505, Security, Steganography, and Watermarking of Multimedia Contents IX, 65050T (27 February 2007)

[18] Güera D, Delp EJ. Deepfake video detection using recurrent neural networks. In2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) 2018 Nov 27 (pp. 1-6). IEEE.