Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

Detecting tampered videos with multimedia forensics and deep learning

Markos Zampoglou[1], Foteini Markatopoulou[1], Gregoire Mercier[2], Despoina Touska[1], Evlampios Apostolidis[1,3], Symeon Papadopoulos[1], Roger Cozien[2], Ioannis Patras[3], Vasileios Mezaris[1], and Ioannis Kompatsiaris[1]

[1] Centre for Research and Technology Hellas, Thermi-Thessaloniki, Greece
{markzampoglou, markatopoulou, apostolid, papadop, bmezaris, ikom}@iti.gr
https://mklab.iti.gr/
[2] eXo maKina, Paris, France
{gregoire.mercier, roger.cozien}@exomakina.fr
http://www.exomakina.fr
[3] School of EECS, Queen Mary University of London, London, UK
I.Patras@qmul.ac.uk

Abstract. User-Generated Content (UGC) has become an integral part of the news reporting cycle. As a result, the need to verify videos collected from social media and Web sources is becoming increasingly important for news organisations. While video verification is attracting a lot of attention, there has been limited effort so far in applying video forensics to real-world data. In this work we present an approach for automatic video manipulation detection inspired by manual verification approaches. In a typical manual verification setting, video filter outputs are visually interpreted by human experts. We use two such forensics filters designed for manual verification, one based on Discrete Cosine Transform (DCT) coefficients and a second based on video requantization errors, and combine them with Deep Convolutional Neural Networks (CNN) designed for image classification. We compare the performance of the proposed approach to other works from the state of the art, and discover that, while competing approaches perform better when trained with videos from the same dataset, one of the proposed filters demonstrates superior performance in cross-dataset settings. We discuss the implications of our work and the limitations of the current experimental setup, and propose directions for future research in this area.

Keywords: Video forensics · video tampering detection · video verification · video manipulation detection · user-generated video.

1 Introduction

With the proliferation of multimedia capturing devices during the last decades, the amount of video content produced by non-professionals has increased rapidly. Respectable news agencies nowadays often need to rely on User-Generated Content (UGC) for news reporting. However, the videos shared by users may not be

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

authentic. People may manipulate a video for various purposes, including propaganda or comedic effect, but such tampered videos pose a major challenge for news organizations, since publishing a tampered video as legitimate news could seriously hurt an organization's reputation. This creates an urgent need for tools that can assist professionals to identify and avoid tampered content.

Multimedia forensics aims to address this need by providing algorithms and systems that assist investigators with locating traces of tampering and extracting information on the history of a multimedia item. Research in automatic video verification has made important progress in the recent past; however, state-of-the-art solutions are not yet mature enough for use by journalists without specialized training. Currently, real-world video forensics mostly rely on expert verification, i.e. trained professionals visually examining the content under various image maps (or *filters*[4]) in order to spot inconsistencies.

In this work, we explore the potential of two such novel filters, originally designed for human visual inspection, in the context of automatic verification. The filter outputs are used to train a number of deep learning visual classifiers, in order to learn to discriminate between authentic and tampered videos. Besides evaluations on established experimental forensics datasets, we also evaluate them on a dataset of well-known tampered and untampered news-related videos from YouTube to assess their potential in real-world settings. Our findings highlight the potential of adapting manual forensics approaches for automatic video verification, as well as the importance of cross-dataset evaluations when aiming for real-world application. A third contribution of this work is a small dataset of tampered and untampered videos collected from Web and social media sources that is representative of real cases.

2 Related Work

Multimedia forensics has been an active research field for more than a decade. A number of algorithms (known as *active* forensics) work by embedding invisible watermarks on images which are disturbed in case of tampering. Alternatively, *passive* forensics aim to detect tampering without any prior knowledge [12]. Image forensics is an older field than video forensics, with a larger body of proposed algorithms and experimental datasets, and is slowly reaching maturity as certain algorithms or algorithm combinations are approaching sufficient accuracy for real-world application. Image tampering detection is often based on detecting local inconsistencies in JPEG compression information, or – especially in the cases of high-quality, low-compression images – detecting local inconsistencies in the high-frequency noise patterns left by the capturing device. A survey and evaluation of algorithms focused on image splicing can be found in [23].

The progress in image forensics might lead to the conclusion that similar approaches could work for tampered video detection. If videos were simply sequences of frames, this might hold true. However, modern video compression

[4] While not all maps are technically the result of filtering, the term *filters* is widely used in the market and will also be used here.

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

is a much more complex process that often removes all traces such as camera error residues and single-frame compression traces [14]. Proposed video forensics approaches can be organized in three categories: double/multiple quantization detection, inter-frame forgery detection, and region tampering detection.

In the first case, systems attempt to detect if a video or parts of it have been quantized multiple times [16, 21]. A video posing as a camera-original User-Generated Content (UGC) but exhibiting traces of multiple quantizations may be suspicious. However, with respect to newsworthy UGC, such approaches are not particularly relevant since in the vast majority of cases videos are acquired from social media sources. As a result, both tampered and untampered videos typically undergo multiple strong requantizations and, without access to a purported camera original, they have little to offer in our task.

In the second category, algorithms aim to detect cases where frames have been inserted in a sequence, which has been consecutively requantized [20, 24]. Since newsworthy UGC generally consists of a single shot, such frame insertions are unlikely to pass unnoticed. Frame insertion detection may be useful for videos with fixed background (e.g. CCTV footage) or for edited videos where new shots are added afterwards, but the task is outside the scope of this work.

Finally, the third category concerns cases where parts of a video sequence (e.g. an object) have been inserted in the frames of another. This the most relevant scenario for UGC, and the focus of our work. Video region tampering detection algorithms share many common principles with image splicing detection algorithms. In both cases, the assumption is that there exists some invisible pattern in the item, caused by the capturing or the compression process, which is distinctive, detectable, and can be disturbed when foreign content is inserted. Some approaches are based solely on the spatial information extracted independently from frames. Among them, the most prominent ones use oriented gradients [17], the Discrete Cosine Transform (DCT) coefficients' histogram [6], or Zernike moments [2]. These work well as long as the video quality is high, but tend to fail at higher compression rates as the traces on which they are based are erased. Other region tampering detection strategies are based on the motion component of the video coding, modeling motion vector statistics [19, 7] or motion compensation error statistics [1]. These approaches work better with still background and slow moving objects, using motion to identify shapes/objects of interest in the video. However, these conditions are not often met by UGC.

Other strategies focus on temporal noise [9] or correlation behavior [8]. The noise estimation induces a predictable feature shape or background, which imposes an implicit hypothesis such as a limited global motion. The *Cobalt* filter we present in Section 3 adopts a similar strategy. The Motion Compensated Edge Artifact is another alternative to deal with the temporal behavior of residuals between I, P and B frames without requiring strong hypotheses on the motion or background contents. These periodic artifacts in the DCT coefficients may be extracted through a thresholding technique [15] or spectral analysis [3]. This approach is also used for inter-frame forgery detection under the assumption that the statistical representativeness of the tampered area should be high.

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

Recently, the introduction of deep learning approaches has led to improved performance and promising results for video manipulation detection. In [22], the inter-frame differences are calculated for the entire video, then a high-pass filter is applied to each difference output and the outputs are used to classify the entire video as tampered or untampered. High-pass filters have been used successfully in the past in conjunction with machine learning approaches with promising results in images [4]. In a similar manner, [13] presents a set of deep learning approaches for detecting face-swap videos created by Generative Adversarial Networks. Besides presenting a very large-scale dataset for training and evaluations, they show that a modified Xception network architecture can be used to detect forged videos on a per-frame basis.

In parallel to published academic work, a separate line of research is conducted by private companies, with a focus on the creation of filters for manual analysis by trained experts. These filters represent various aspects of video content, including pixel value relations, motion patterns, or compression parameters, and aim at highlighting inconsistencies in ways that can be spotted by a trained person. Given that tools based on such filters are currently in use by news organizations and state agencies for judiciary or security reasons, we decided to explore their potential for automatic video tampering detection, when using them in tandem with deep learning frameworks.

3 Methodology

The approach we explore is based on a two-step process: forensics-based feature extraction and classification. The feature extraction step is based on two novel filters, while the classification step is based on a modified version of the GoogLeNet and ResNet deep Convolutional Neural Networks (CNN).

3.1 Forensics-based filters

The filters we used in our experiments, originally designed to produce visible output maps that can be analyzed by humans, are named *Q4* and *Cobalt*. The *Q4* filter is used to analyze the decomposition of the image through the Discrete Cosine Transform (DCT). It is applied on each individual video frame, irrespective of whether they are I, P, or B frames. Each frame is split into $N \times N$ blocks (typically $N = 8$), and the two-dimensional DCT is applied to transform each image block into a block of the same size in which the coefficients are identified based on their frequency. The first coefficient (0,0) represents low frequency information, while higher coefficients represent higher frequencies. JPEG compression takes place in the YCbCr color spectrum, and we then use the Y channel (luminance) for further analysis.

If we transform all $N \times N$ blocks of a single image band with the DCT, we can build $N \times N$ (e.g. 64 for JPEG) different coefficient arrays, each one using a single coefficient from every block - for example, an image of the coefficients (0,0) of each block, and a different one using the coefficients (0,1). Each one of the $N \times N$

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

coefficient arrays has size equal to $1/N$ of the original image in each dimension. An artificially colorized RGB image may then be generated by selecting 3 of these arrays and assigning each to one of the three RGB color channels. This allows us to visualize three of the DCT coefficient arrays simultaneously, as well as the potential correlation between them. Combined together, the images from all frames can form a new video of the same length as the original video, which can then be used for analysis. The typical block size for DCT (e.g. in JPEG compression) is 8×8. However, analysis in 8×8 blocks yields too small coefficient arrays. Instead, 2×2 blocks are used so that the resulting output frame is only half the original size. A selection of coefficients (0, 1), (1, 0) and (1, 1) generates the final output video map of the Q4 filter.

The second filter we use is the *Cobalt* filter. This compares the original video with a modified version of it, re-quantized using MPEG-4 at a different quality level (and a correspondingly different bit rate). If the initial video contains a (small) area that comes from another stream, this area may have undergone MPEG-4 quantization at a level which is different from the original one. This area may remain undetectable by any global strategy attempting to detect multiple quantization. The principle of the Cobalt filter is straightforward: We requantize the video and calculate the per-pixel values, creating an *Error Video*, i.e. a video depicting the differences. In theory, if we requantize using the exact same parameters that were used for the original video, there will be almost no error to be seen. As the difference with the original increases, so does the intensity of the error video. In designing Cobalt, a "compare-to-worst" strategy has been investigated, i.e. if a constant quality encoding is done, the comparison will be performed with the worst possible quality, and conversely, if a constant bit rate encoding is done, the comparison is performed with the worst possible bit rate. This induces a significantly contrasted video of errors when the quantization history of the initial video is not homogeneous.

3.2 Filter output classification

Both filters produce outputs in the form of RGB images. Following the idea that the filter maps were originally intended to be visually evaluated by a human expert, we decided to treat the problem as a visual classification task. This allows us to combine the maps with Convolutional Neural Networks pre-trained for image classification. Specifically, we take an instance of GoogLeNet [18] and an instance of ResNet [5], both pre-trained on the ImageNet classification task, and adapt them to the needs of our task. The image outputs from the filtering process are scaled to match the default input size of the CNNs, i.e. 224×224 pixels. In contrast to other forensics-based approaches, where rescaling might destroy sensitive traces, the filters we use are aimed for visual interpretation by humans, so -as in any other classification task- rescaling should not cause problems.

To improve classification performance, the networks are extended using the method of [11], according to which adding an extra Fully Connected (FC) layer

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

prior to the final FC layer can improve performance when fine-tuning a pre-trained network. In this case, we added a 128-unit FC layer to both networks, and also replaced the final 1000-unit FC layer with a 2-unit layer, since instead of the the 1000-class ImageNet classification task, here we are dealing with a binary (tampered/untampered) task. As the resulting networks are designed for image classification, we feed the filter outputs to each network one frame at a time, during both training and classification.

4 Experimental Study

4.1 Datasets and Experimental Setup

The datasets we used for our study came from two separate sources. One comprised the Development datasets provided by the NIST 2018 Media Forensics Challenge[5] for the *Video Manipulation Detection* task. There are two separate development datasets, named Dev1 and Dev2, the first consisting of 30 video pairs (i.e. 30 tampered videos and their 30 untampered sources), and the second of 86 video pairs, containing approximately 44K and 134K frames respectively. The task also included a large number of distractor videos, which were not included in our experiments. These two datasets, Dev1 and Dev2, are treated as independent sets, but since they originate from the same source, they likely exhibit similar features.

The second source of videos was the InVID Fake Video Corpus [10], developed over the course of the InVID project. The Fake Video Corpus (FVC) contains 110 real and 117 fake newsworthy videos from social media sources, which include not only videos that have been tampered but also videos that are contextually false (e.g. whose description on YouTube contains misinformation about what is shown). Out of that dataset, we selected 35 real videos that comprise single-shot, unedited videos, and 33 fake videos that include tampered UGC, but no explicit edit effects such as cuts or text overlay because these would be easy to detect compared to the single-shot untampered videos. This dataset contains 163K frames, equally split between tampered and untampered videos.

The temporal annotation of the datasets is incomplete, i.e. we do not always know where and when the tampering takes place, only that a video contains a tampered part. For our experiments, we treated all frames originating from untampered videos as untampered, and all frames originating from tampered videos as tampered. This is a weak assumption that will surely also lead to erroneous annotations. Based on manual inspection, we found that this assumption does hold true for the majority of videos, and thus we consider the quality of annotations adequate for the task.

Experimental Setup. In order to evaluate the proposed approaches, the two filters, Q4 and Cobalt, were applied to all frames of the videos, and then each of

[5] https://www.nist.gov/itl/iad/mig/media-forensics-challenge-2018

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

Fig. 1. Indicative videos from the FVC dataset. Top (tampered videos): "Bear attacks cyclist", "Lava selfie", "Bear attacks snowboarder", "Eagle drops snake". Bottom (untampered videos): "Stockholm attack", "Hudson landing", "Istanbul attack" and "Giant aligator in golf field".

the two networks was trained with each filter output independently on the task of distinguishing between tampered and untampered frames. For comparison, we also implemented the following video features from the state of the art:

- *rawKeyframes* [13]. The video is decoded into its frames and the raw keyframes (without any filtering process) are given as input to the deep network.
- *highPass frames* [4]. The video is decoded into its frames, each frame is filtered by a high-pass filter and the filtered frame is given as input to the deep network.
- *frameDifference* [22]. The video is decoded into its frames, the frame difference between two neighboring frames is calculated, the new filtered frame is also processed by a high-pass filter and the final filtered frame is given as input to the deep network.

The filter outputs are used to train the networks. During evaluation, for each video the arithmetic mean of the classification scores for all of its frames is calculated separately for each class (tampered, untampered). The video is classified as tampered if the average score for the tampered class is larger than the average score for the untampered class. Experiments were run both by training and evaluating on the same dataset (using 5-fold cross-validation) and by training and testing on different datasets to evaluate each algorithm's ability to generalize. In all cases, we used three performance measures: Accuracy, Mean Average Precision (MAP), and Mean Precision for the top-20 retrieved items (MP@20).

4.2 Within-dataset Experiments

Preliminary evaluations of the proposed approach took the form of within-dataset evaluations, using five-fold cross-validation. We used the two datasets from the NIST Challenge (Dev1 and Dev2), as well as their union, for these runs. The results are presented in Table 1.

The results show that, for all filters and models, Dev1 is significantly more challenging. Accuracy for all cases ranges between 0.58 and 0.68, while the same

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

Table 1. Within-dataset evaluations

Dataset	Filter-DCNN	Accuracy	MAP	MP@20
Dev1	cobalt-gnet	**0.6833**	0.7614	-
	cobalt-resnet	0.5833	0.6073	-
	q4-gnet	0.6500	**0.7856**	-
	q4-resnet	0.6333	0.7335	-
Dev2	cobalt-gnet	0.8791	**0.9568**	**0.8200**
	cobalt-resnet	0.7972	0.8633	0.7600
	q4-gnet	**0.8843**	0.9472	0.7900
	q4-resnet	0.8382	0.9433	0.7600
Dev1 + Dev2	cobalt-gnet	**0.8509**	0.9257	0.9100
	cobalt-resnet	0.8217	0.9069	0.8700
	q4-gnet	0.8408	**0.9369**	**0.9200**
	q4-resnet	0.8021	0.9155	0.8700

measure for Dev2 ranges from 0.79 to 0.88. Mean Average Precision follows a similar pattern. It should be noted that the MP@20 measure does not apply to Dev1 cross-validation due to its small size (the test set would always contain less than 20 items).

Merging the two datasets gives us the largest cross-validation dataset set from which we can expect the most reliable results. In terms of Accuracy and MAP, for Dev1+Dev2 the results are slightly worse than Dev2, and significantly better than Dev1. MP@20 is improved compared to Dev2 but this can be attributed to the relatively small size of Dev2. Overall, the results appear encouraging, reaching a Mean Average Precision of 0.94 for the Dev1+Dev2 set. GoogLeNet seems to generally perform better than ResNet. In terms of performance, the two filters appear comparable, with Cobalt outperforming Q4 at some cases, and the inverse being true for others.

4.3 Cross-dataset Experiments

Using the same dataset or datasets from the same origin for training and testing is a common practice in evaluations in the field. However, as in all machine learning tasks, the machine learning algorithm may end up picking up features that are characteristic of the particular datasets. This means that the resulting model will be unsuitable for real-world application. Our main set of evaluations concerns the ability of the proposed algorithms to deal with cross-dataset classification, i.e. training the model on one dataset and testing it on another. We used three datasets: Dev1, Dev2, and FVC. We run three sets of experiments using a different dataset for training each time. One set was run using Dev1 as the training set, the second using Dev2, and the third using their combination. Dev1 and Dev2 originate from the same source, and thus, while different, may exhibit similar patterns. Thus, we would expect that training on Dev1 and evaluating on Dev2 or vice versa would be easier than evaluating on FVC.

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

Table 2. Cross-dataset evaluations (Train: Dev1)

Training	Testing	Filter-DCNN	Accuracy	MAP	MP@20
Dev1	**Dev2**	cobalt-gnet	0.6033	0.8246	0.9000
		cobalt-resnet	0.6364	0.8335	0.9000
		q4-gnet	0.5124	0.8262	0.9000
		q4-resnet	0.5041	0.8168	0.9000
		rawKeyframes-gnet [13]	0.5868	0.8457	0.8500
		rawKeyframes-resnet [13]	0.2893	0.6588	0.4000
		highPass-gnet [4]	0.5620	0.8134	0.8500
		highPass-resnet [4]	0.5537	0.7969	0.8000
		frameDifference-gnet [22]	0.6942	**0.8553**	**0.9000**
		frameDifference-resnet [22]	**0.7190**	0.8286	0.8500
	FVC	cobalt-gnet	0.4412	0.3996	0.3000
		cobalt-resnet	0.4706	0.5213	0.5000
		q4-gnet	0.58824	0.6697	0.6000
		q4-resnet	**0.6029**	**0.6947**	**0.7000**
		rawKeyframes-gnet [13]	0.5294	0.5221	0.5000
		rawKeyframes-resnet [13]	0.5147	0.4133	0.2500
		highPass-gnet [4]	0.5441	0.5365	0.5000
		highPass-resnet [4]	0.5000	0.5307	0.6000
		frameDifference-gnet [22]	0.5735	0.5162	0.4500
		frameDifference-resnet [22]	0.5441	0.4815	0.5000

The results are shown in Tables 2, 3, and 4. As expected, evaluations on the FVC dataset yield relatively lower performance than evaluations on Dev1 and Dev2. In terms of algorithm performance, a discernible pattern is that, while the state of the art seems to outperform the proposed approaches on similar datasets (i.e. training on Dev1 and testing on Dev2 or vice versa), the Q4 filter seems to outperform all other approaches when tested on FVC. Specifically, *frameDifference* from [22] clearly outperforms all competing approaches when cross-tested between Dev1 and Dev2. However, its performance drops significantly when evaluated on the FVC dataset, indicating an inability to generalize to different, –and in particular, real-world– cases. This is important, since in real-world application and especially in the news domain, the data will most likely resemble those of the FVC dataset (i.e. user-generated videos). It is unlikely that we will be able to collect enough videos to train a model so that it knows the characteristics of such videos beforehand.

The Q4 filter reaches a MAP of 0.71 when trained on the combination of the Dev1 and Dev2 datasets, and tested on the FVC dataset. This performance, while significantly higher than all alternatives, is far from sufficient for application in newsrooms. It is, however, indicative of the potential of the specific filter. Another observation concerns the choice of networks. While in most experiments there was no clear winner between GoogLeNet and ResNet, it seems that the former performs better on average, and consistently better or comparably to ResNet when tested on the FVC dataset.

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

Table 3. Cross-dataset evaluations (Train: Dev2)

Training	Testing	Filter-DCNN	Accuracy	MAP	MP@20
Dev2	**Dev1**	cobalt-gnet	0.6167	0.6319	0.6500
		cobalt-resnet	0.5333	0.7216	0.6000
		q4-gnet	**0.6500**	0.7191	**0.7000**
		q4-resnet	0.5833	0.6351	0.6000
		rawKeyframes-gnet	**0.6500**	0.6936	0.6500
		rawKeyframes-resnet	0.6333	0.6984	0.6500
		highPass-gnet [4]	0.5667	0.6397	0.6500
		highPass-resnet [4]	**0.6500**	0.6920	**0.7000**
		frameDifference-gnet [22]	0.6167	**0.7572**	**0.7000**
		frameDifference-resnet [22]	**0.6500**	0.7189	**0.7000**
	FVC	cobalt-gnet	0.5588	0.5586	0.5500
		cobalt-resnet	0.5000	0.4669	0.4000
		q4-gnet	**0.6177**	**0.6558**	0.7000
		q4-resnet	0.5147	0.4525	0.4000
		rawKeyframes-gnet [13]	0.5147	0.6208	**0.7000**
		rawKeyframes-resnet [13]	0.5735	0.6314	0.6500
		highPass-gnet [4]	0.4706	0.5218	0.4500
		highPass-resnet [4]	0.5588	0.5596	0.6000
		frameDifference-gnet [22]	0.5000	0.5652	0.6000
		frameDifference-resnet [22]	0.5000	0.5702	0.6500

5 Conclusions and Future Work

We presented our efforts in combining video forensics filters, originally designed to be visually examined by experts, with deep learning models for visual classification. We explored the potential of two forensics-based filters combined with two deep network architectures, and observed that, while for training and testing on similar videos the proposed approach performed comparably or worse than various state of the art filters, when evaluated on different datasets than the ones used for training, one of the proposed filters clearly outperformed all others. This is an encouraging result that may reveal the potential of such an approach towards automatic video verification, and especially for content originating from web and social media.

However, the current methodology has certain limitations that should be overcome in the future for the method to be usable in real settings. One is the problem of annotation. During our experiments, training and testing was run on a per-frame basis, in which all frames from tampered videos were treated as tampered, and all frames from untampered videos as untampered. This assumption is problematic, as a tampered video may also contain untampered frames. However, as we lack strong, frame-level annotation, all experiments were run using this weak assumption. For the same reason, the final classification of an entire video into "tampered" or "untampered" was done by majority voting. This may also distort results, as it is possible that only a few frames of a video have been tampered, and yet this video should be classified as tampered.

Proc. 25th Int. Conf. on Multimedia Modeling (MMM 2019), Springer LNCS vol. 11295. Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-030-05710-7_31

Table 4. Cross-dataset evaluations (Train: Dev1+Dev2)

Training	Testing	Filter-DCNN	Accuracy	MAP	MP@20
Dev1 + Dev2	**FVC**	cobalt-gnet	0.4706	0.4577	0.4000
		cobalt-resnet	0.4853	0.4651	0.4500
		q4-gnet	**0.6471**	**0.7114**	**0.7000**
		q4-resnet	0.5882	0.6044	0.6500
		rawKeyframes-gnet	0.5882	0.5453	0.5000
		rawKeyframes-resnet	0.5441	0.5175	0.5500
		highPass-gnet	0.5294	0.5397	0.5500
		highPass-resnet	0.5441	0.5943	0.6000
		frameDifference-gnet	0.5441	0.5360	0.6000
		frameDifference-resnet	0.4706	0.5703	0.5500

The limitations of the current evaluation mean that the results can only be treated as indicative. However, as the need for automatic video verification methods increases, and since the only solutions currently available on the market are filters designed for analysis by experts, the success of such filters using automatic visual classification methods is strongly encouraging. In the future, we aim to improve the accuracy of the approach in a number of ways. One is to improve the quality of the dataset by adding temporal annotations for tampered videos, in order to identify which frames are the tampered ones. Secondly, we intend to develop a larger collection of state-of-the-art implementations on video tampering detection, to allow for more comparisons. Finally, we will explore more nuanced alternatives to the current voting scheme where each video is classified as tampered if more than half the frames are classified as such.

Acknowledgements

This work is supported by the InVID project, which is funded by the European Commissions Horizon 2020 program under contract number 687786.

References

1. Chen, S., Tan, S., Li, B., Huang, J.: Automatic detection of object-based forgery in advanced video. IEEE Trans. on Circuits Systems and Video Technologies **26**(11), 2138–2151 (Nov 2016)
2. D'Amiano, L., Cozzolino, D., Poggi, G., Verdoliva, L.: Video forgery detection and localization based on 3d patchmatch. In: IEEE Int. Conf. on Multimedia expo Workshop (ICMEW) (2015)
3. Dong, Q., Yang, G., Zhu, N.: A MCEA based passive forensics scheme for detecting frame based video tampering. Digital Investigation pp. 151–159 (2012)
4. Fridrich, J., Kodovsky, J.: Rich models for steganalysis of digital images. IEEE Transactions on Information Forensics and Security **7**(3), 868–882 (2012)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

6. Labartino, D., Bianchi, T., Rosa, A.D., Fontani, M., Vazquez-Padin, D., Piva, A.: Localization of forgeries in MPEG-2 video through gop size and dq analysis. In: IEEE Int. Workshop on Multimedia and Signal Processing. pp. 494–499 (2013)
7. Li, L., Wang, X., Wang, G., Hu, G.: Detecting removed object from video with stationary background. In: Proc. of the 11th Int. Conf. on Digital Forensics and Watermarking (WDW). pp. 242–252 (2013)
8. Lin, C.S., Tsay, J.J.: A passive approach for effective detection and localization of region-level video forgery with spatio-temporal coherence analysis. Digit. Investig. **11**(2), 120–140 (2014)
9. Pandey, R., Singh, S., Shukla, K.: Passive copy-move forgery detection in videos. In: IEEE Int. Conf. Computer and Comm. Technology (ICCCT). pp. 301–306 (2014)
10. Papadopoulou, O., Zampoglou, M., Papadopoulos, S., Kompatsiaris, Y., Teyssou, D.: Invid Fake Video Corpus v2.0 (version 2.0). Dataset on Zenodo (2018)
11. Pittaras, N., Markatopoulou, F., Mezaris, V., Patras, I.: Comparison of fine-tuning and extension strategies for deep convolutional neural networks. In: International Conference on Multimedia Modeling. pp. 102–114. Springer (2017)
12. Piva, A.: An overview on image forensics. ISRN Signal Processing pp. 1–22 (2013)
13. Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., Nießner, M.: Faceforensics: A large-scale video dataset for forgery detection in human faces. arXiv preprint arXiv:1803.09179 (2018)
14. Sitara, K., Mehtre, B.M.: Digital video tampering detection: An overview of passive techniques. Digital Investigation **18**, 8–22 (2016)
15. Su, L., Huang, T., Yang, J.: A video forgery detection algorthm based on compressive sensing. Multimedia Tools and Applications **74**, 6641–6656 (2015)
16. Su, Y., Xu, J.: Detection of double compression in mpeg-2 videos. In: IEEE 2nd International Workshop on Intelligent Syetems and Application (ISA) (2010)
17. Subramanyam, A., Emmanuel, S.: Video forgery detection using hog features and compression properties. In: IEEE 14th Int. Workshop on Multimedia and Signal Processing (MMSP). pp. 89–94 (2012)
18. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
19. Wang, W., Farid, H.: Exposing digital forgeries in interlaced and deinterlaced video. IEEE trans. on Information Forensics and Security **2**(3), 438–449 (2007)
20. Wu, Y., Jiang, X., Sun, T., Wang, W.: Exposing video inter-frame forgery based on velocity field consistency. In: ICASSP (2014)
21. Xu, J., Su, Y., liu, Q.: Detection of double MPEG-2 compression based on distribution of dct coefficients. Int. J. Pattern Recognition and A.I. **27**(1) (2013)
22. Yao, Y., Shi, Y., Weng, S., Guan, B.: Deep learning for detection of object-based forgery in advanced video. Symmetry **10**(1), 3 (2017)
23. Zampoglou, M., Papadopoulos, S., Kompatsiaris, Y.: Large-scale evaluation of splicing localization algorithms for web images. Multimedia Tools and Applications p. online first (2016). https://doi.org/10.1007/s11042-016-3795-2
24. Zhang, Z., Hou, J., Ma, Q., Li, Z.: Efficient video frame insertion and deletion detection based on inconsistency of correlations between local binary pattern coded frames. Security and Communication networks **8**(2) (25 January 2015)