https://doi.org/10.2352/EI.2024.36.4.MWSF-335

Efficient Temporally-Aware DeepFake Detection using H.264 Motion Vectors

Peter Grönquist*, Yufan Ren*, Qingyi He, Alessio Verardo, Sabine Süsstrunk
Image and Visual Represenation Lab, École Polytechnique Fédérale de Lausanne; Lausanne, Switzerland
**** Equal Contribution***

Abstract

Video DeepFakes are fake media created with Deep Learning (DL) that manipulate a person's expression or identity. Most current DeepFake detection methods analyze each frame independently, ignoring inconsistencies and unnatural movements between frames. Some newer methods employ optical flow models to capture this temporal aspect, but they are computationally expensive. In contrast, we propose using the related but often ignored Motion Vectors (MVs) and Information Masks (IMs) from the H.264 video codec, to detect temporal inconsistencies in DeepFakes. Our experiments show that this approach is effective and has minimal computational costs, compared with per-frame RGB-only methods. This could lead to new, real-time temporally-aware DeepFake detection methods for video calls and streaming.

Introduction

Ever since their emergence, DeepFakes, or more specifically Deep Learning (DL)-based fake media with manipulated facial identity and expression, have brought serious security and privacy concerns to the public [5, 43]. Whilst they offer advantages to photo and video editing, they also have significant downsides, including identity abuse and the spreading of misinformation. Ranging from defamation and political misuse to video-call scams and pornography, these issues have become a major cause for concern in our society [14]; even more so as facial identity is a determining factor for person recognition, both in the human thought process and in machine algorithms. [17]. Given their importance in our social interactions [16], a simple modification of facial expressions can have profound effects in how a person is perceived.

Therefore, to combat DeepFake videos, various research groups, companies and organizations have launched multiple campaigns to raise public awareness of these issues [2, 6]. These initiatives have notably accelerated research on DeepFake countermeasures, demonstrating significant improvements in accuracy, efficiency, and generalizability [13, 34, 33, 27, 36, 47, 32, 11, 12, 1, 4, 9]. However, despite these achievements, most contemporary DeepFake detection methods exhibit two key limitations: the omission of temporal video information and insufficient generalizability [48].

With the fast-paced evolution of DeepFake generation models, it is necessary for DeepFake detection algorithms to be more robust to these unknown manipulations, as measured by cross-forgery accuracy. Current such attempts at improving generalizability focus mostly on finding new fake cues that are as algorithm-independent as possible, such as detecting heartbeat rhythms [32, 11, 12] or blending boundaries [27]. However, these

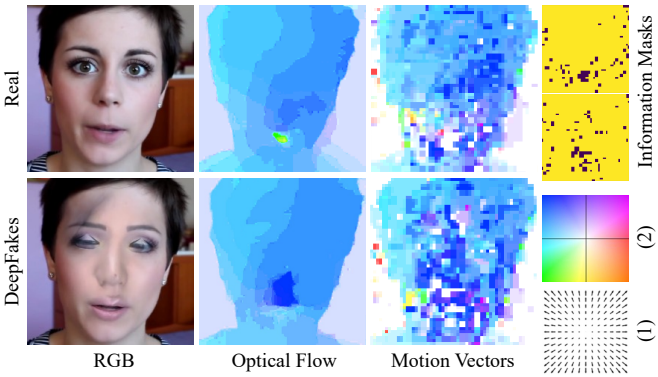

Figure 1. *Two continuous video frames in the FaceForensics++ dataset, with their corresponding optical flow and H.264 motion vectors, with respect to the precursory frame of each frame. The information masks indicate the availability of motion vectors at a spatial location. (1) and (2) express the color visualization of two dimensional motion information into RGB space. Motion vectors show similar motion as optical flow but are coarser and noisier. (Best viewed on a screen when zoomed in)*

come with their own restrictions of needing a specific quality of videos or not having the video fully generated.

Handling temporal information poses its own specific challenges; a typical DeepFake detection pipeline will sample multiple frames from a video, predict per-frame fake probabilities, and then heuristically aggregate these probabilities into an overall fake video probability. This method, however, fails to account for the inherent temporal consistency stemming from real-world constraints, such as stable facial features, unchanged eye colors, and naturally paced blinking. One common way of capturing this temporal information is to use the motion information in videos, commonly represented as Optical Flow (OF) [9]. However, one issue is that optical flow estimation requires additional computational resources, in a sequential manner, which poses a potential efficiency bottleneck. This is especially the case considering the increasingly growing prevalence of DeepFakes in live streaming and online video platforms.

Based on the two above-mentioned limitations, this paper proposes the use of H.264 motion vectors as a method for motion approximation in DeepFake detection: On one hand, H.264 motion vectors offer new and unique fake artifacts and MV-based models are shown to be more generalizable than per-frame RGB-only methods. On the other hand, we do not need to estimate motion vectors as in optical flow-based methods, which is a benefit provided by the widespread adoption of the H.264 codec. In research on using MV for segmentation [38], the motion approx-

imation capabilities of MVs and OF are compared. They suggest that MVs can provide a good estimate for optical flow, except for small noise, which can for example stem from imperceptible variations in luminosity. In those cases, MVs might reference different patches, similar in luminosity and color, that are contrary to the optical flow, or generally just plain noisy. It is exactly these temporal inconsistencies, we hypothesize, that might present a challenge for generative DeepFake methods, and which we can exploit as an auxiliary input for detection.

To keep model performance high on top of keeping computational costs low, we propose a classification model based on MobileNet [21, 20]. To demonstrate the effectiveness of our framework, we conduct rigorous experiments on the FaceForensics++ dataset [33], which consists of several different types of DeepFake generated content. In these experiments, and compared with our optical flow baseline classifier, our motion vector based model achieves a relative improvement of $\sim$14% in accuracy. We summarize our findings with the following contributions:

- We propose a novel DeepFake detection framework that takes into account temporal transformations and artifacts, outperforming state-of-the-art optical flow based models.
- We reach real-time data extraction efficiencies by using readily available H.264 motion vectors as motion approximation.
- We demonstrate that our proposed method achieves higher generalization capabilities than models based solely on RGB input, and rival combined RGB and optical flow based models.

Related work

DeepFakes Detection. Based on whether they use the inter-frame information, we categorize DeepFakes detection algorithms into two types: image-level and temporally-aware. Image-level detections exploit intra-frame information, such as forensics features [13], XceptionNet features [34, 33], and blending boundary [27, 36]. However, image-level algorithms fail to use temporal cues in videos, such as cross-frame inconsistency [47], heartbeat rhythms [32, 11, 12], phoneme-viseme mismatching [1], and unnatural movement [4, 9].

Temporally-aware DeepFake Detection. To exploit temporal information, researchers proposed and adapted deep networks that can process multiple frames from the computer vision community, such as 3DCNNs [48], spatiotemporal transformers [47], recurrent networks [19], and LSTMs [3]. Comparatively, instead of feeding all frames as concatenated input to the same network, research by Simonyan and Zisserman claims there is an advantage in using two-stream networks. These networks that have two branches for encoding inputs, feed in pre-extracted motion information, i.e. optical flow, in a separate branch from the RGB input, and have for example been found to help with action classification [37]. Optical flows, in this case, describe object motion, and we consider it as one of the most general forms of motion representation for DeepFake detection [4, 9].

One issue that might however hinder the use of optical flow for DeepFake detection is the computational cost of optical flow estimation. Despite the efforts that have been made to speed up optical flow estimation to near real-time or real-time[1] [25, 39], the computational costs in DeepFake detection are still quite large, as OF has to be calculated for every single frame of a video. Therefore it is beneficial to further reduce the resource usage. In this paper, instead of trying to speed up optical flow estimation, we use the almost free motion vectors in the H.264 video codec as an approximation to the optical flow for DeepFakes detection.

Motion vectors as motion approximation. H.264 uses motion vectors to exploit temporal redundancy in video frames for compression purposes. Motion vectors bear similarity to the optical flow, in that they are both two-dimensional data describing pixel or patch level motion across frames. This similarity and the motion approximation relation has been used for a long time, for example in video object detection, researchers use motion vector to propagate object detection result of key frames [45]. Motion vectors have also been used to represent a compressed version of videos, that can be used as input directly [44], reducing the data size by up to two orders of magnitude. In our case, we use the motion itself as a discriminative feature for DeepFake detection, by similarly making use of the compressed temporal information.

Preliminaries and Methodology

In this Section, we first briefly introduce optical flow, and motion vectors. After that, we discuss our data processing methods in Section, as well as the classifier. The whole detection pipeline is illustrated in Fig. 2.

Optical flow

Optical flows represent the projection of every 3D point's trace to the image plane. The motion information they provide is very helpful for other computer vision tasks, such as detecting and tracking objects [35], and visual odometry [29]. However, extracting optical flow from a video stream is one of computer vision's unsolved tasks, due to its ill-posed nature. Examples include having to deal with occlusion of objects, non-rigid movement or even blurry or noisy images.

Over time there have been several proposed solutions to optical flow estimation, for example, nowadays classical optical flow estimations algorithms solve the ill-posed aperture problem by introducing a smoothness prior, such as the wildly used optical flow estimation algorithm TV-L1 [31, 46]. Researchers also employ more recent deep learning technology to exploit learnable and generalizable priors, achieving state-of-the-art performance on multiple optical flow estimation datasets such as Sintel [8] and KITTI 2015 [28]. Therefore, for a fair comparison with our methods, we use the RAFT model which has a top score on both of these datasets, as our optical flow baseline [39].

H.264 motion vectors

Motion Vectors (MVs) are part of the H.264[2] video compression scheme that exploits temporal and spatial redundancy for a better compression rate. Unlike optical flow which gives a movement prediction for each pixel, motion vectors operate on macroblocks, consisting of 4x4, 8x8 and 16x16 pixel blocks, allowing for mixed sizes such as 4x8 and 16x8.

To fully understand the computational advantage of using encoded MVs, we shortly recap their encoding and decoding pro-

[1] https://developer.nvidia.com/opticalflow-sdk

[2] https://www.itu.int/rec/T-REC-H.264-202108-I/en

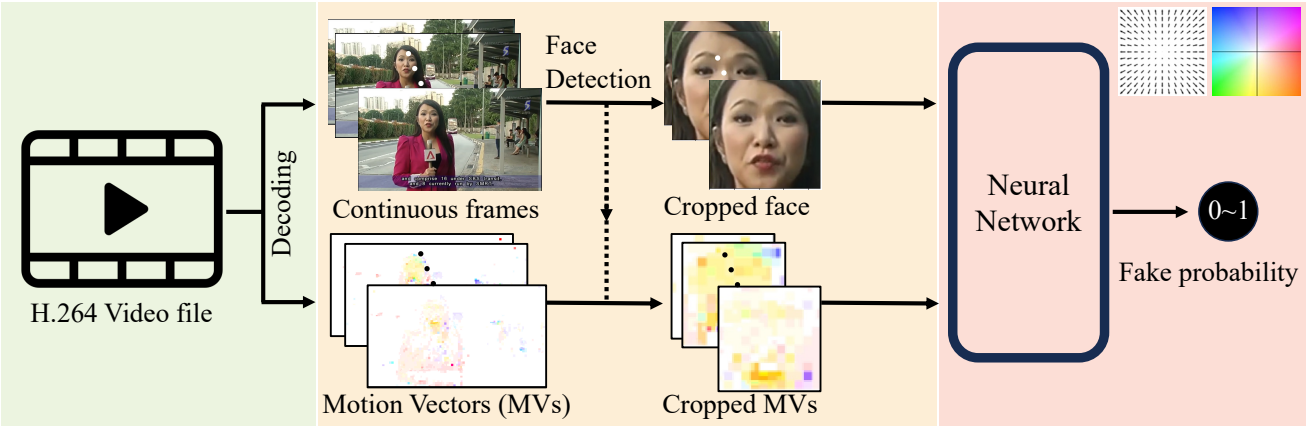

***Figure 2. Proposed DeepFake detection pipeline**. The input to the processing set-up is an H.264 encoded video-file or stream. The file or stream are then further processed by selecting a subset of the RGB-frames and cropping the face via a face-detection network. During the decoding process, the motion vectors and a mask of the I-macroblocks are taken directly from the decoder, cropped to the bounding box and used as input to either the standalone MobileNet, or our Two-Stream network. The final inputs to the networks vary depending on the transformations performed on the motion vectors and the network used. For example we can use temporal slices of varying duration, here multiple frames, alone, with RGB, and with the I-macroblock mask. Information masks are omitted for simplicity. (Best viewed on a screen when zoomed in)*

cess. There exist three types of frames in the H.264 format: Intra-coded (I), Predicted (P), Bi-directionally predicted (B). Of the three, the I-frames do not interest us, as they contain no temporal information and are fully intra-coded. The P- and B-frames both reference other frames for their encoding and decoding, with the difference being that P-frames reference only past frames and B-frames reference past and future frames. Both are however limited to a maximal distance of 16 frames from the current frame. Therefore, we can gain temporal information, here MVs, from P- and B-frames.

In P- and B-frames, MVs are not always available for all macro-blocks. Some macro-blocks are encoded entirely with intra-frame coding, so called I-Macroblocks. We use a binary mask to represent these and call the result our Information Mask (IM), meaning there is no temporal dependency in those macroblocks and all information is *new*. We do not delve deeper into how the I-Macroblocks are coded, as only the encoder's incapability of encoding the information with MVs interests us.

Another interesting process is how this temporal information is encoded and decoded. For this the H.26X compression schemes follow a motion estimation algorithm: Firstly, MVs between the to-be-encoded macroblocks are predicted. This can be done via diverse easy-to-compute mechanisms (e.g. median of previous, in encoding order, neighbours) and vary depending on the H.26X version. Then, the actual MV v is calculated for the respective macroblock and the MV estimation from the previous step is subtracted from it, leading to the Motion Vector Differences (MVD). This is then the information that will be sent over the bitstream and then decoded. The decoding process happens in reverse, meaning the MVD are recovered, the MVs are estimated, and v, our target MVs, are calculated by adding the two.

Knowing that the encoding process can be quite computationally expensive, hardware encoding and decoding solutions that automatize this process and completely remove the CPU-load, have become quite prevalent. There are even GPU hardware based decoding acceleration to support this process to allow for a faster decoding process without extra hardware. From there we surmise that using a hardware accessory that directly outputs the motion vectors into memory, it would be *quasi-free*, in terms of CPU and GPU processing power, to get the MV information. However, as a proof of concept, we perform our following experiments using non-hardware-specific video decoding, as the source of the MVs should have no impact on the results.

Data Preprocessing

Face detector. Following [9, 27], we use a pretrained MTCNN [49] face detector to detect the face region using the default parameters. Should multiple faces be detected, only the biggest one is used. After detecting the four corner points, the dimension that is smallest is then padded with pixels to obtain a square bounding box. The face is resized to a 224×224 resolution. Finally the resulting frames are normalized using standardization.

Motion Vectors. After processing the faces, we extract the MVs and IMs in the face region and stack them together. We then end up utilizing either a four-dimensional input (past-x, past-y, future-x, future-y) to indicate whether the MV references a past or future frame and their direction, or a six-dimensional input (future- and past-referencing IMs) if the IM is employed as an additional input feature. The MVs are also normalized by standardization.

There is however a caveat, in that there exist blocks which have a zero-MV, and blocks for which there is no MV information, that are also zero. As these non MV-blocks are represented by the IM we can exclude them from the normalization by referencing non-zero IM blocks. The MV and IM blocks are then scaled up to our selected input resolution, by using bi-linear and nearest neighbour interpolation respectively.

Data Augmentation. We apply several augmentations, to both reduce overfitting and give us better predictions, inspired by the DFDC winner [6]. We use the albumentation library [7] to implement the following augmentations, which are then applied with a certain probability each (see Figure 3): image compression, gaussian noise and blur, RGB and hue-saturation-value

Figure 3. *Examples of all the data augmentations performed on RGB data stemming from FaceForensics++.*

shifts, FancyPCA [24], random Brightness Contrast, and grey-scale transformation.

On top of these augmentations, we apply non-RGB transformations to the entirety of the image, which include horizontal and vertical flips, as well as the complete removal of certain regions of the input, replacing them by patches of zeroes as introduced in GridMask data augmentation [10]. It is to note that flips and GridMask augmentations affect MVs differently, as their values need to be mirrored in vector space on top of flipping their position, which can be seen in Figure 4.

Classifier

We choose MobileNetV3 [21, 20] as our classifier's backbone. MobileNetV3 is designed using neural architecture search for low latency and accuracy, aligning with our goal of wanting to make our pipeline as efficient as possible. To use it for DeepFake detection, we made several modifications. Firstly, we replace the classifier output by a fully connected layer to output a scalar indicating the probability of the sample being fake. Secondly, we change the number of input channels of the first convolution, as we have varying channel sizes, depending on whether we use RGB, MVs, or MVs and IMs. The feature extraction part remains the same. Finally for our two-stream network, we combine the two different modalities of input, RGB and MVs by concatenating the last layers of their respective MobileNets, which consist of a single value. We then get our final prediction by averaging 100 randomly sampled frames of the video.

Loss function. DeepFake detection, as we define it, is a binary classification, and therefore we use the binary cross entropy loss.

$$\mathscr{L} = \hat{y}\log y + (1-\hat{y})\log(1-y), \tag{1}$$

where $\hat{y}$ and y denote the predicted and ground-truth label.

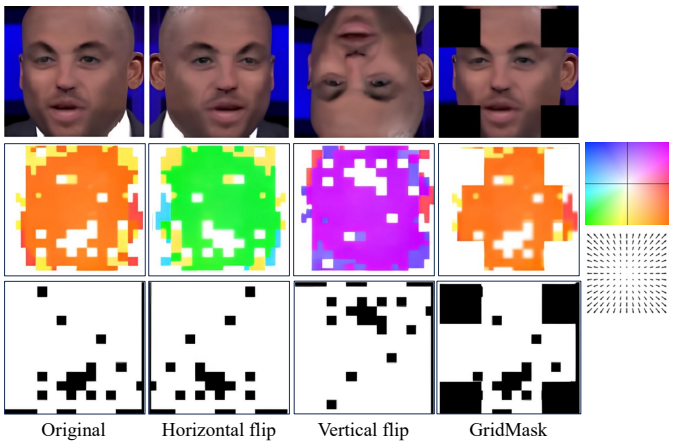

Figure 4. *Examples of the data augmentations performed on motion vectors and information mask on the FaceForensics++ dataset. In these augmentations, the motion vector values have to be additionally mirrored on the flip axis in vector space, on top of flipping their positions. The colors represent the MVs direction and length in the vector field. (Best viewed on a screen when zoomed in)*

Experiments

Experimental setups

Datasets. Following previous optical flow based DeepFake detection research [9], we report our findings on the FaceForensics++ [33] dataset, which contains 1,000 YouTube videos that have been manipulated with different types of DeepFake manipulations, we use the hq version (C23). More specifically there are five different types: FaceShifter (FS) [26], FaceSwap (FSwap) [23], DeepFakes (DF) [5], Face2Face (F2F) [41], and NeuralTexture (NT) [40]. The diversity of the data, and the pairing of fake and non-fake video of the same source, allow for practical initial explorations into generalizability of DeepFake detection methods, which is our goal.

Baselines. To fairly evaluate the performance of our motion vector based model, we compare it not only with pure RGB based models, but also with a state-of-the-art optical flow estimator, RAFT [39], based model, using the default parameters as described in the original paper. We create our own baselines, as we cannot compare to values from [9], which does not report all quantitative values. Additionally, RAFT significantly outperforms TV-l1, the classical OF algorithm they use[3]. To avoid storing floating points, which would make for very large file sizes, we save the estimated optical flows as gray scale images using lossless jpeg compression as in [37].

Implementation details. We implement our model in PyTorch [30] and PyTorch Lightning [15]. The training and experiments are performed on two Nvidia GTX Titan X with 12 GB of memory and an Intel Xeon E5-2680 v3 @ 2.50GHz CPU. We use the Adam optimizer [22] and perform a balanced training, meaning depending on the dataset, the diverse DeepFake generated contents are sampled equally and sum up to the number of real samples. The models are trained for eight epochs (until convergence) and the best performing checkpoint on validation loss is selected.

[3] `http://sintel.is.tue.mpg.de/results`

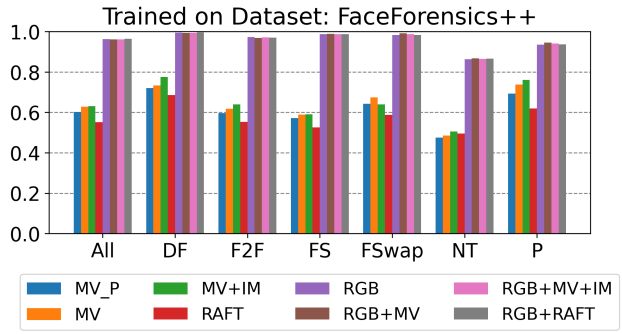

Figure 5. *General evaluation of all our models on all FaceForensics++ DeepFake types. The values represent the accuracies on the respective testsets. We also examine MV_P, which represents the use of only past frame referencing motion vectors, which would make it more closely matched to OF methods. P represents the pristine (real) videos themselves. The values quickly overfit when combined with RGB, and adding auxiliary inputs only changes the accuracy by fractions of percentages.*

DeepFake detection accuracy

To gain an understanding of their general accuracy in DeepFake detection, we evaluate the two-stream models for our temporally-aware models on the full dataset, and compare it to the RGB baseline (see Figure 5). The goal being to make the prediction as accurate as possible.

Once we introduce RGB based models into the mix, we see an immediate saturation in accuracy for the models at around 96%. Additionally, the RGB models once again quickly overfit on the dataset. As we see no changes in general accuracy whether the model uses RGB or a combination of RGB and motion information, we surmise that this is due to the poor quality of current DeepFake datasets available for research. Meaning the model gets as high of an accuracy it can get, already by just using RGB input.

To have a fairer evaluation of these combined models, we stress the importance of introducing better DeepFake detection datasets that include high quality real-world data.

Cross-forgery generalization ability

A main focus in our experiments is to check whether we can maintain the generalizability that was initially shown when using optical flow models, over RGB models [9]. To that end, we train our models on one type of DeepFake each, and evaluate them on all different DeepFake types in the dataset, as shown in Figure 6, Table 3 and Table 4. Firstly, and as we elaborate in Section , we can see MVs achieving higher accuracies on the specific datasets, indicating that they might add additional information that is not included in optical flows. Secondly, for cross-forgery and on average we see equally strong generalization results for all temporal augmentations, with OF performing slightly better on some subsets, namely Face2Face and DeepFakes. Confirming previous research, we see purely RGB trained networks perform poorly on all generalization tasks.

Thereby, we confirm the cross-forgery detection ability of MV in DeepFake detection and perform a more in-depth comparison of their similarities in subsection .

Classification with only temporal data

In this experiment we want to evaluate the performance of the classifier when fed with only temporal information. Previous research [9] reports average values of 82.99% for this task, however due to the different model, optical flow estimator and test set, we cannot use these numbers as direct baseline. Therefore we establish our own baseline by using the state-of-the-art RAFT optical flow as input and compare it with MVs and IMs concatenated with MVs (see Figure 5 and Table 1). We also evaluate the behavior of models relying solely on MVs stemming from the past (MV_P), which would correspond more closely to the OF.

These results show that using the MVs and IMs only, strongly outperforms the RAFT based model, even only using MV_Ps. Meaning that, while maybe not all motion is accounted for by MVs as it is by the OF (see Section), there is additional information in MVs and IMs that allows for a better classification of whether a video is a forgery or not.

Method	DF	F2F	FS	FSwap	NT
OF	67.90	66.00	64.13	63.30	61.37
MV_P	71.97	66.90	72.93	71.00	71.63
MV	77.60	69.50	68.20	77.93	69.53
MV+IM	**83.53**	**76.50**	**75.30**	**81.17**	**74.23**

Evaluation of motion vectors, information masks and optical flow classification accuracies (in %) on the respective DeepFake types they were trained on.

Resolution	OF MFLOP	MV MFLOP
480×360	$138.8 \cdot 10^3$	0.1
640×480	$249.3 \cdot 10^3$	0.2
1280×720	$783.5 \cdot 10^3$	0.6
1920×1080	$1.9 \cdot 10^6$	1.3

Computational costs of creating optical flow based on RAFT compared with the costs of transforming the MVs into our input format, in MFLOPs.

Run-time analysis

We perform a general upper-bound analysis of the estimated operation cost of preparing motion vectors, as compared to obtaining the optical flow. We do this by firstly calculating the costs of transforming the motion vectors to fit our input type, assuming the encoded data is provided for free, and comparing it with RAFT FLOPs obtained by fvcore[4] given two input frames. Two frames as the OF algorithm needs to predict the motion between the two. The different computational costs are visible in Table 2.

These values represent the difference between a single frame, therefore one can easily see that the processing time scales massively with a regular 30 fps stream from a webcam.

Using MVs as motion approximation

Beyond the capabilities of MVs in DeepFake generalization, we also want to confirm the validity of MVs for motion approximation, which has already been used in prior research [45]. To evaluate their efficacy in this task, we assessed MVs on the Sintel optical flow benchmark [8]. Using the clean data split for simplicity, we generated MVs by compressing video frames into a

[4] https://github.com/facebookresearch/fvcore

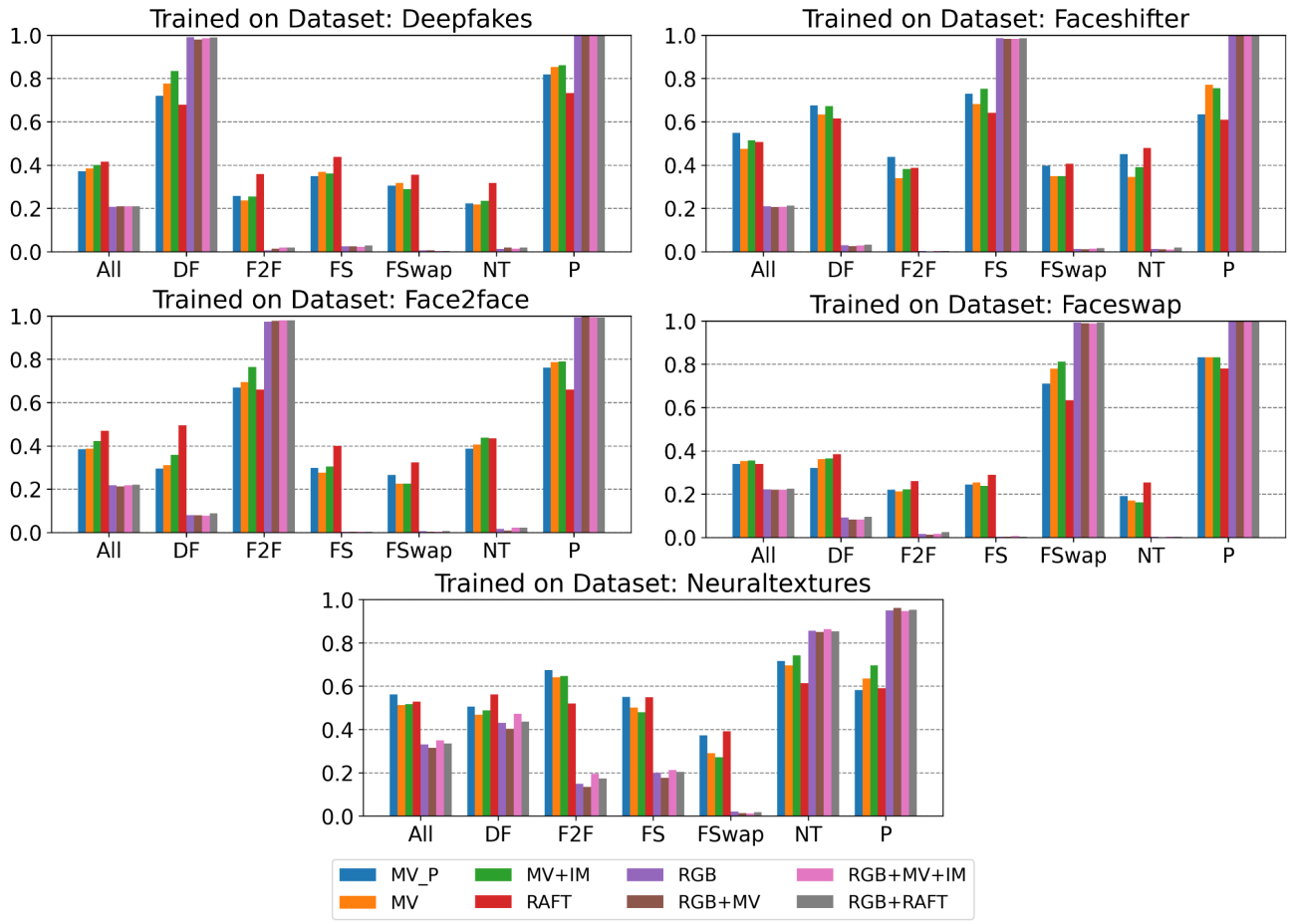

Figure 6. *Cross-forgery evaluation of our models on datasets they were not trained for (out-of-distribution) in the FaceForensics++ dataset. The values represent the accuracies [0.0,1.0] of the models on the respective DeepFake or real (pristine) video test set.*

MV+IM	DeepFakes	Face2Face	FaceShifter	FaceSwap	NeuralTextures	all
DeepFakes	83.53%	35.77%	67.23%	36.47%	48.73%	**77.57%**
Face2Face	25.57%	76.50%	38.23%	22.20%	64.60%	63.93%
FaceShifter	36.10%	30.40%	75.30%	23.83%	47.87%	59.00%
FaceSwap	28.97%	22.57%	34.90%	81.17%	27.20%	64.00%
NeuralTextures	23.43%	43.63%	39.07%	16.23%	**74.23%**	50.47%
Pristine	**86.10%**	**78.93%**	**75.63%**	**83.17%**	69.63%	76.03%
all	40.10%	42.10%	51.43%	35.53%	51.70%	63.03%

Cross-forgery accuracy evaluation of our MobileNet network with MV+IM input, trained on a specific DeepFake type (columns). Evaluated on different forgery types (rows).

RAFT	DeepFakes	Face2Face	FaceShifter	FaceSwap	NeuralTextures	all
DeepFakes	67.90%	49.47%	61.53%	38.43%	56.20%	**68.60%**
Face2Face	35.83%	66.00%	38.73%	25.97%	51.93%	55.23%
FaceShifter	43.63%	40.00%	**64.13%**	28.97%	54.80%	52.57%
FaceSwap	35.43%	32.50%	40.63%	63.30%	39.13%	58.77%
NeuralTextures	31.73%	43.47%	47.87%	25.43%	**61.37%**	49.53%
Pristine	**73.33%**	**66.07%**	61.00%	**78.00%**	59.03%	61.90%
all	41.53%	46.87%	50.60%	34.00%	52.83%	55.13%

Cross-forgery accuracy evaluation of our MobileNet network with RAFT input as baseline, trained on a specific DeepFake type (columns). Evaluated on different forgery types (rows).

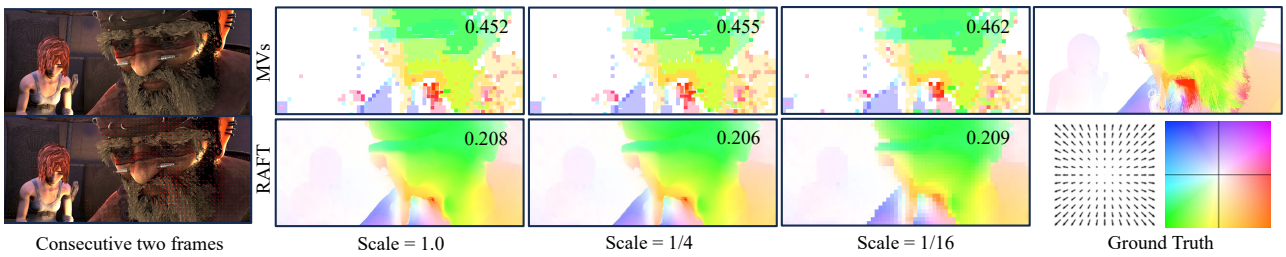

Figure 7. *Comparison of MVs and optical flow, values indicates end-point-error shown. Image brightness is corrected for better visualization. (Best viewed on a screen when zoomed in)*

video via FFmpeg [42] using the H.264 codec. Subsequently, in line with the optical flow evaluation, we computed the error of the MVs and tabulated the results in table 5.

We note that our computation only incorporated the P frames, as I frames lack motion information. We report end-point-error (EPE) in the table for different resolutions. We compare the different scales, as the MVs are per definition at a 16 times lower resolution. While the EPE results show significantly worse results for MV, they represent valid motion approximations by surpassing some of the current existing OF on OF benchmarks [18]. It should also be taken into consideration that RAFT was trained on the Sintel dataset.

DownScale	Method	EPE ↓
1/1	RAFT	**0.603**
	MVs	2.193
1/4	RAFT	**0.611**
	MVs	2.364
1/16	RAFT	**0.652**
	MVs	2.527

Evaluation of MVs as motion approximation. End-point-error (EPE) is reported.

Ablation on data augmentation

We run an ablation study on different types of data augmentation, and evaluate their contribution individually in Table 6. Training without augmentations led to quickly overfitting models, and adding augmentations generally gave us a performance and generalization boost.

Augmentations	Accuracy
None	93.05%
+ Compression, Noise, Blur	89.34%
+ Color changes	93.62%
+ GridMask	93.51%
+ Flips	94.62%

Augmentations we use and their accuracy on the test set, when used with our two-stream model consisting of RGB+MV+IM streams.

Limitations and future work

There are two noteworthy limitations in our method. First, we propose to replace optical flow with H.264 motion vectors for DeepFakes detection. Though it reduces computational costs by estimating the optical flow, we still have two efficiency bottlenecks in our pipeline: Firstly we still need to run face detection and classifier inference sequentially. Although we note that it would be possible to further exploit MVs for face detection and face tracking to boost prediction speed [45, 44]. Secondly, as MVs have a 16 times lower resolution than the source video, it becomes significantly more challenging for the classifier when the potential subject of forgery does not occupy a larger part of the frame, or the frame has a low resolution to begin with. A solution we see, would be a compromise: Using MVs as priors to guide the estimation of optical flow as in [44]. Since the coarse motion information would is provided by MVs, the optical flow estimation network can have a lighter design.

While we show these capabilities for the most widely adapted H.264 codec, they are just as well portable to the newer codecs, such as H.265, which uses the Coding Tree Unit (CTU) structure instead of macroblocks, but still retains motion vectors.

Conclusion

In this work we have shown that the information available through the H.264 encoding-decoding process, can directly be used to augment existing RGB-only DeepFake detection pipelines, by including temporal information and artifacts. This allows for better generalization capabilities, comparable with optical flow based methods. Furthermore the temporal information is made available at a vastly reduced cost and can remove the need of running an optical flow network sequentially, before being able to run the rest of the detection pipeline. Finally, by leveraging hardware-based solutions for the decoding process, this auxiliary information becomes quasi-free. This, in turn, would enable real-time temporal anomaly detection even on consumer-grade hardware, highlighting its potential value in various applications such as video calls and streaming settings.

References

[1] Shruti Agarwal et al. “Detecting deep-fake videos from phoneme-viseme mismatches”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2020, pp. 660–661.

[2] AI_Singapore. “Trusted Media Challenge, Accessed: 2023-07-15”. In: `https://trustedmedia.aisingapore.org/` (2021).

[3] Irene Amerini and Roberto Caldelli. “Exploiting prediction error inconsistencies through LSTM-based classifiers to detect deepfake videos”. In: *Proceedings of the 2020 ACM workshop on information hiding and multimedia security*. 2020, pp. 97–102.

[4] Irene Amerini et al. “Deepfake video detection through optical flow based cnn”. In: *Proceedings of the IEEE/CVF international conference on computer vision workshops*. 2019, pp. 0–0.

[5] Anonymous. “Deepfakes, Accessed: 2023-07-15”. In: *https://github.com/ondyari/FaceForensics/tree/master/\\dataset/DeepFakes* (2018).

[6] Microsoft AWS Facebook. “Deepfake Detection Challenge, Accessed: 2023-07-15”. In: *https://www.kaggle.com/c/deepfake-detection-challenge/overvi/* (2021).

[7] Alexander V. Buslaev et al. “Albumentations: fast and flexible image augmentations”. In: *CoRR* abs/1809.06839 (2018). arXiv: 1809.06839. URL: http://arxiv.org/abs/1809.06839.

[8] D. J. Butler et al. “A naturalistic open source movie for optical flow evaluation”. In: *European Conf. on Computer Vision (ECCV)*. Ed. by A. Fitzgibbon et al. (Eds.) Part IV, LNCS 7577. Springer-Verlag, Oct. 2012, pp. 611–625.

[9] Roberto Caldelli et al. “Optical Flow based CNN for detection of unlearnt deepfake manipulations”. In: *Pattern Recognition Letters* 146 (2021), pp. 31–37.

[10] Pengguang Chen et al. “GridMask Data Augmentation”. In: *CoRR* abs/2001.04086 (2020). arXiv: 2001.04086. URL: https://arxiv.org/abs/2001.04086.

[11] Umur Aybars Ciftci, Ilke Demir, and Lijun Yin. “Fakecatcher: Detection of synthetic portrait videos using biological signals”. In: *IEEE transactions on pattern analysis and machine intelligence* (2020).

[12] Umur Aybars Ciftci, Ilke Demir, and Lijun Yin. “How do the hearts of deep fakes beat? deep fake source detection via interpreting residuals with biological signals”. In: *2020 IEEE international joint conference on biometrics (IJCB)*. IEEE. 2020, pp. 1–10.

[13] Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva. “Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection”. In: *Proceedings of the 5th ACM workshop on information hiding and multimedia security*. 2017, pp. 159–164.

[14] Jesse Damiani. “A Voice Deepfake Was Used To Scam A CEO Out Of $243,000, Accessed: 2023-07-15”. In: *https://www.forbes.com/sites/jessedamiani/2019/09/03/\\a-voice-deepfake-was-used-to-\\scam-a-ceo-out-of-243000/* (2019).

[15] William Falcon and The PyTorch Lightning team. *PyTorch Lightning*. Version 1.4. Mar. 2019. DOI: 10.5281/zenodo.3828935. URL: https://github.com/Lightning-AI/lightning.

[16] Chris Frith. “Role of facial expressions in social interactions”. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 364.1535 (2009), pp. 3453–3458.

[17] Javier Galbally, Sébastien Marcel, and Julian Fierrez. “Biometric antispoofing methods: A survey in face recognition”. In: *IEEE Access* 2 (2014), pp. 1530–1552.

[18] Mathias Gehrig et al. “E-RAFT: Dense Optical Flow from Event Cameras”. In: *International Conference on 3D Vision (3DV)*. 2021.

[19] David Güera and Edward J Delp. “Deepfake video detection using recurrent neural networks”. In: *2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS)*. IEEE. 2018, pp. 1–6.

[20] Andrew Howard et al. “Searching for mobilenetv3”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 1314–1324.

[21] Andrew G Howard et al. “Mobilenets: Efficient convolutional neural networks for mobile vision applications”. In: *arXiv preprint arXiv:1704.04861* (2017).

[22] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[23] Marek Kowalski. “FaceSwap, Accessed: 2023-07-15”. In: *https://github.com/ondyari/FaceForensics/tree/master/\\dataset/FaceSwapKowalski* (2016).

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[25] Till Kroeger et al. “Fast Optical Flow using Dense Inverse Search”. In: *CoRR* abs/1603.03590 (2016). arXiv: 1603.03590. URL: http://arxiv.org/abs/1603.03590.

[26] Lingzhi Li et al. “Advancing high fidelity identity swapping for forgery detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 5074–5083.

[27] Lingzhi Li et al. “Face x-ray for more general face forgery detection”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 5001–5010.

[28] Moritz Menze, Christian Heipke, and Andreas Geiger. “Object Scene Flow”. In: *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)* (2018).

[29] Peter Muller and Andreas Savakis. “Flowdometry: An optical flow and deep learning based approach to visual odometry”. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2017, pp. 624–631.

[30] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[31] Javier Sánchez Pérez, Enric Meinhardt-Llopis, and Gabriele Facciolo. “TV-L1 optical flow estimation”. In: *Image Processing On Line* 2013 (2013), pp. 137–150.

[32] Hua Qi et al. “Deeprhythm: Exposing deepfakes with attentional visual heartbeat rhythms”. In: *Proceedings of the 28th ACM international conference on multimedia*. 2020, pp. 4318–4327.

[33] Andreas Rossler et al. “Faceforensics++: Learning to detect manipulated facial images”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 1–11.

[34] Andreas Rössler et al. “Faceforensics: A large-scale video dataset for forgery detection in human faces”. In: *arXiv preprint arXiv:1803.09179* (2018).

[35] Jeongho Shin et al. “Optical flow-based real-time object tracking using non-prior training active feature model”. In: *Real-time imaging* 11.3 (2005), pp. 204–218.

[36] Kaede Shiohara and Toshihiko Yamasaki. “Detecting Deepfakes with Self-Blended Images”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 18720–18729.

[37] Karen Simonyan and Andrew Zisserman. “Two-stream convolutional networks for action recognition in videos”. In: *Advances in neural information processing systems* 27 (2014).

[38] C. Solana-Cipres et al. “Real-time moving object segmentation in H.264 compressed domain based on approximate reasoning”. In: *International Journal of Approximate Reasoning* 51.1 (2009), pp. 99–114. ISSN: 0888-613X. DOI: https://doi.org/10.1016/j.ijar.2009.09.002. URL: https://www.sciencedirect.com/science/article/pii/S0888613X0900139X.

[39] Zachary Teed and Jia Deng. “Raft: Recurrent all-pairs field transforms for optical flow”. In: *European conference on computer vision*. Springer. 2020, pp. 402–419.

[40] Justus Thies, Michael Zollhöfer, and Matthias Nießner. “Deferred neural rendering: Image synthesis using neural textures”. In: *ACM Transactions on Graphics (TOG)* 38.4 (2019), pp. 1–12.

[41] Justus Thies et al. “Face2face: Real-time face capture and reenactment of rgb videos”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2387–2395.

[42] Suramya Tomar. “Converting video formats with FFmpeg”. In: *Linux Journal* 2006.146 (2006), p. 10.

[43] Luisa Verdoliva. “Media forensics and deepfakes: an overview”. In: *IEEE Journal of Selected Topics in Signal Processing* 14.5 (2020), pp. 910–932.

[44] Chao-Yuan Wu et al. “Compressed video action recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 6026–6035.

[45] Takanori Yokoyama, Toshiki Iwasaki, and Toshinori Watanabe. “Motion vector based moving object detection and tracking in the MPEG compressed domain”. In: *2009 Seventh International Workshop on Content-Based Multimedia Indexing*. IEEE. 2009, pp. 201–206.

[46] Christopher Zach, Thomas Pock, and Horst Bischof. “A duality based approach for realtime tv-l 1 optical flow”. In: *Joint pattern recognition symposium*. Springer. 2007, pp. 214–223.

[47] Daichi Zhang et al. “Deepfake video detection with spatiotemporal dropout transformer”. In: *arXiv preprint arXiv:2207.06612* (2022).

[48] Daichi Zhang et al. “Detecting Deepfake Videos with Temporal Dropout 3DCNN.” In: *IJCAI*. 2021, pp. 1288–1294.

[49] Kaipeng Zhang et al. “Joint face detection and alignment using multitask cascaded convolutional networks”. In: *IEEE signal processing letters* 23.10 (2016), pp. 1499–1503.

Author Biography

Peter Grönquist *is a Machine Learning Engineer at EPFL's Image and Visual Representation Lab. Starting his research at ETHZ in 2016, he specialized in computer vision at Huawei Research Zürich. At EPFL, he delves into image and video generation and detection. Besides academia, he consults with industry on DeepFake detection and promotes tech awareness and accessibility.*

Yufan Ren *is a direct Ph.D. student (2020-) at EPFL's Image and Visual Representation Lab. He focuses on 3D vision and Neural Rendering and graduated from Zhejiang University, receiving the Chu Kochen Award.*

Qingyi He *is a MSc student in Computer Science at EPFL and is performing research in DeepFake analysis at the IVRL. She received her BSc in Computer Science from Zhejiang university.*

Alessio Verardo *is a MSc student in Data Science at EPFL and has performed research on video DeepFake detection within IVRL. He received his BSc in communication systems at EPFL in 2021.*

Sabine Süsstrunk *has directed EPFL's Image and Visual Representation Lab since 1999 and was the inaugural Director of the Digital Humanities Institute from 2015-2020. Specializing in computational photography, machine learning, and image quality. She's chaired numerous international conferences and is the President of the Swiss Science Council SSC, with roles in EPFL-WISH, SRG SSR, Largo Films and is a Fellow of IEEE and IS&T.*