POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

A CNN-based detector for video frame-rate interpolation

Laurea Magistrale in Music and Acoustic Engineering - Ingegneria Musicale e Acustica

Author: Simone Mariani

Advisor: Prof. Paolo Bestagini

Academic year: 2021-2022

---

1. Introduction

In the last twenty years, due to the increasing popularity of social media networks and to the technological advancements in video capturing devices, videos are spread everywhere. Anyone can capture videos with extreme facility, but unfortunately, anyone can also tamper with them. Maliciously manipulated videos can lead to several consequences like people defamation, fake-news spreading or mass opinion formation. For this reason, it is important to guarantee the integrity and the authenticity of the video contents, and this is exactly the aim of our work.
Among the many video manipulation operations that are available, video frame-rate interpolation is one of the most common ones. This operation consists in creating new frames (upsampling) in between already existing consecutive frames, or even in dropping (downsampling) some frames, resulting in a frame-rate variation. If new frames are created, they can be a repetition of the original ones, or they can be built by interpolating between the original frames pixels. The interpolating approach is very difficult to detect, especially if the motion is taken into account during the interpolation, for example using Motion Compensated Interpolation (MCI) [1].
Indeed, MCI approaches perform first a motion estimation step, and then an interpolation one. The idea is to estimate the trajectories that each pixel follow from one frame to another. Then, the pixels of the newly generated frames are placed on the trajectory indicated by the estimation. In this way the interpolated frames are computed and inserted in the middle of the original frames, resulting in a very fluid sequence.
Since video frame-rate interpolation can lead to very realistic results that may be used for malicious video tampering, it is necessary to develop some tools capable of identifying manipulations introduced by interpolation.
The goal of our work, is to be able to classify a given video, as frame-rate interpolated or as original (untouched frame-rate).
With this purpose, we propose a detector capable to identify traces of frame-rate interpolation in a video.
The rest of the document is organized in the following way.
In Section 2 we formally define the goal of our work. In Section 3 we sum up the details of our detector, and we also introduce how to extend the approach to localize smaller interpolated video regions. Section 4 is devoted to present the principal results obtained from the conducted experiments. In the final Section 5, we draw the conclusions of our work.

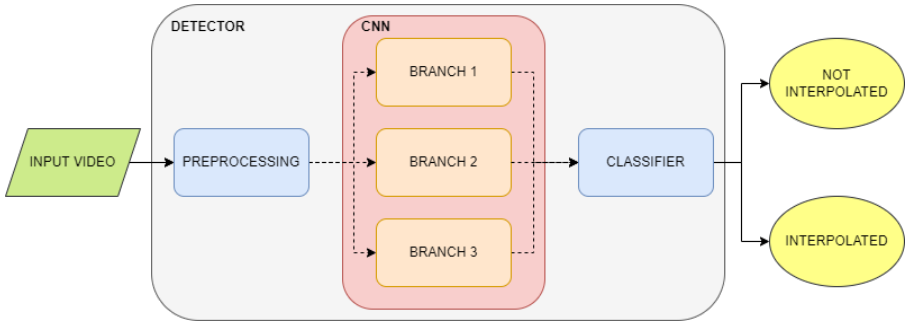

Figure 1: A quick overview of our detector. We analyze the input video in order to classify it as interpolated or not interpolated.

2. Problem Formulation

Video frame-rate interpolation leaves peculiar footprints on the newly generated frames. It is therefore possible to study these traces in a forensic manner in order to detect if the video has been interpolated in time or not.

The goal of this work is to detect if a video under analysis has undergone frame-rate interpolation or not. With reference to Figure 1, this is a binary classification problem, with one video as input, and one binary label as output.

Formally, let us consider a video sequence defined as

$$\mathsf{V} = [\mathsf{F}_0; \ldots; \mathsf{F}_{L-1}]; \quad (1)$$

where F_i; i 2 [0; L 1] denotes the i-th frame in V and L represents the number of frames contained in V. Our goal is to learn how to associate a label p to the video V where p = 1 means that the video has been temporally interpolated, whereas p = 0 means that the video frame-rate is untouched.

3. Proposed Method

To detect frame-rate interpolation in a video we propose a method structured as indicated in Figure 1. This is composed by three principal stages:

1. Preprocessing
2. Convolutional Neural Network (CNN)
3. Classifier

Preprocessing In the Preprocessing block, we extract T + 1 consecutive frames from the video, producing a segment S. Then we convert all the frames of S in the grayscale colorspace creating the segment S^{gray}. After that, we resize all of the frames of the video segment S^{gray} according to our policy. In particular, in the training phase of our detector, the frames are first resized by a random factor r picked between 0.9 and 1.1, in a way to have a final height of r H and final width of r W, being H and W the original frames height and weight. Then a center crop of N N is extracted producing the segment $\mathsf{S}_{N\times N}$. In the testing phase, instead, a simple center crop of N N is extracted from S^{gray} producing the preprocessed segment $\mathsf{S}_{N\times N}$. The final step in the preprocessing stage is to produce three different versions of the segment $\mathsf{S}_{N\times N}$, this to try to capture different information about the possible interpolation traces present in the input video V. These three versions are S_{fra} (original frames), S_{res} (residuals) and S_{opt} (optical flow), all these segments are of length T frames and all of their frames are of dimension N N.

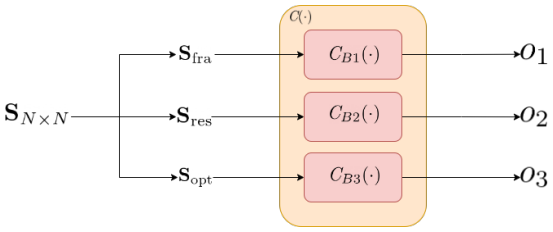

Figure 2: CNN high level structure. From the preprocessed segment $\mathsf{S}_{N\times N}$ we obtained the three input segments S_{fra}, S_{res} and S_{opt}. We fed each one of them into the correspondent branch. Three output scores are produced from each branch analysis (o_1, o_2 and o_3).

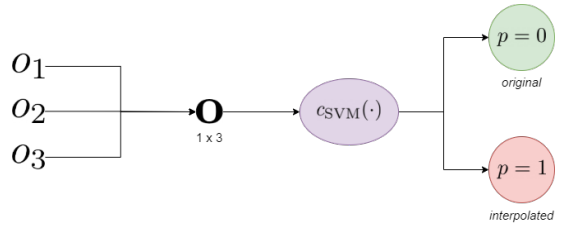

Figure 3: Pipeline of the final classification stage. The three branches outcomes (o_1, o_2 and o_3) are aggregated into the feature vector O. This vector is fed to the classifier $c_{SVM}(\,)$ which predicts if the video is interpolated ($p = 1$) or original ($p = 0$).

CNN The goal of the CNN inside our detector is to process the three segments (S_{fra}, S_{res} and S_{opt}) created during preprocessing and to extract some sort of interpolation traces inside of them.
In Figure 2 this approach is summarized. In particular, the CNN is composed by three structurally identical branches ($C_{B1}(\,)$, $C_{B2}(\,)$ and $C_{B3}(\,)$). Each one of them is devoted to the processing of each one of the preprocessed segments (S_{fra}, S_{res} and S_{opt}). After the processing, for each branch, an output score is produced (o_1, o_2 and o_3). This scores span the range $(\quad 1\,;+1\,)$, the higher the score, the higher the probability that the video has undergone frame-rate interpolation.

Classifier The last stage of our detector has the aim of producing a prediction p related to the input video V by analyzing the CNN branches output scores.
First we want to pack together the o_1, o_2 and o_3 scores in the feature vector O, as indicated in Figure 3.
Then, the idea is to use this feature vector to state if the video has been frame-rate interpolated or not. To do that, we implemented a Support Vector Machine (SVM), in particular a Support Vector Classifier. The feature vector O is fed to the classifier ($c_{SVM}(\,)$), which generates the prediction p, where $p = 0$ means that the video is classified as original, instead $p = 1$ means that the video is classified as interpolated.

To conduct our experiments, first we had to train and test the proposed detector. To do so, we built a new dataset composed by a mixture of original and interpolated videos. Our idea was to start from the Kinetics400 dataset, and to construct a dataset D in which we put some of the Kinetics400 original videos and the correspondent interpolated versions at different frame-rates. All of the videos are interpolated through FFMPEG [2] with a MCI interpolation. The training phase is divided into the training of the individual CNN branches and into the training of the global model (considering the classifier).

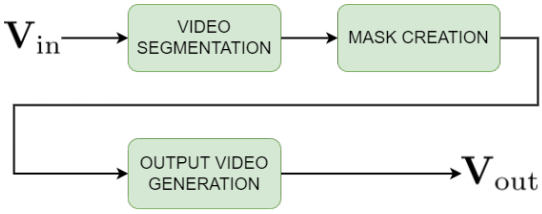

Figure 4: High level pipeline of localization application. From the input video V_{in} to the output video V_{out} in which the interpolated sub blocks are highlighted.

3.1. Localization

In this section we show how to use the proposed detector to localize the interpolated zones inside a video.
The idea is to follow the pipeline presented in Figure 4, starting from the input video V_{in} and producing the output video V_{out} in which the interpolated regions are highlighted. First, we divide the input video V_{in} in segments of length $T + 1$ frames. What we want to do is to analyze each segment in order to highlight the zones in which our detector finds interpolation traces. With this purpose, we divide each segment into multiple subsegments representing non overlapping $N \quad N$ spatial regions of the segment itself. Then, we want to generate predictions related to each subsegment, in a way to mark each subsegment as interpolated or not interpolated. Following this procedure, we preprocess all of the subsegments belonging to each segment in the exact same way as explained in the previous section (in particular, using the testing resizing policy). After that, for each subsegment (of spatial dimension $N \quad N$), we produce the three versions S_{fra}, S_{res} and S_{opt}. Then, for each subsegment, the three created segments are processed through the CNN and the output scores

are produced. The scores are packed together and fed to the classifier which generates a prediction on the subsegment (p=0 means that the subsegment is predicted as original, otherwise p=1 means that the subsegment is predicted as interpolated).
The idea is to collect the predictions for all of the subsegments belonging to a segment, composing a binary mask related to the segment. This binary mask has the same spatial dimension of a frame belonging to the segment. The mask is composed by "zeros" in correspondence of the subsegments predicted as interpolated, instead is composed by "ones" in correspondence of the subsegments predicted as not interpolated. Once this mask is created for every segment belonging to the input video V_{in}, we proceed with the output video generation.
The output video consists in a version of the input video in which the interpolated zones are highlighted. To do this, we rely on the segment masks which represent exactly the interpolated sectors inside of each segment (N N sectors where the mask is set to 1). For each segment, we multiply the segment mask for the green and blue channel of all the frames belonging to the segment. In this way, the resulting segments will be composed by the original frames in the non interpolated zones, instead by only red channel frames in the interpolated zones. Infact, in the interpolated zones (where the segment mask is set to zero) the multiplication suppress green and blue channels of the original frames. The output video V_{out} is then produced by concatenating all the highlighted segments.

4. Results

In this section, we sum up the main results obtained in the two applications of our detector: detection and localization of frame-rate interpolation.

In Table 1 we compare the detection testing accuracies of our model, against two of the most recent works in the video interpolation field. These two works are SpeedNet [3] and the detector proposed by Hosler and Stamm [4].
As we can see from Table 1, our method is very efficient in detecting frame-rate interpolation and outperforms the other two.

METHODS COMPARISON		
Proposed	*SpeedNet* [3]	*S&H [4]*
99.630%	75.600%	91.700%

Table 1: Comparison between our detector against "SpeedNet" [3] and "Stamm and Hosler" detector (S&H) [4] in terms of accuracy.

In Figure 5 we show a result related to the localization application of the proposed detector. We took an input original video and we produced a locally interpolated version of it by replacing the upper left 224x224 block with the correspondent block taken from the interpolated version of the video. Then, we analyzed this video in the way explained in the previous section, dividing it in 16 frames segments, and scanning each segment through subsegments of 224x224 spatial dimension. We produced an output video in which the interpolated sectors for each segment are highlighted in red. In Figure 5, three frames from three different segments have been extracted from the output video. It is evident that the localization is very accurate in this specific case.

5. Conclusions

In this work, we considered the problem of identifying traces of frame-rate interpolation in video sequences. To this purpose we proposed a detector which is trained in a supervised fashion to understand if a video has been frame-rate interpolated. The proposed methodology exploits the idea of applying different kinds of preprocessing to the video under analysis in order to better expose frame-rate interpolation traces. Preprocessed videos are then passed to a CNN to extract salient features, and an SVM that performs the final classification.

Our method has been validated through a series of experiments. To conduct our experiments, we built a brand new dataset composed by original videos and the correspondent interpolated (at different frame-rates) versions, starting from the Kinetics400 dataset. We have demonstrated the precision of our model, evaluating it from multiple perspectives. Moreover, we have compared against two recently-proposed state-of-the-art techniques, showing that the proposed method is able to outperform both.