Realtime Deepfake Detection using Video Vision Transformer

Abhishek Doshi
COE - CNDS
Veermata Jijabai Technological Institute
Mumbai
abhishekdoshi100@gmail.com

Abhinav Venkatadri
COE - CNDS
Veermata Jijabai Technological Institute
Mumbai
abhinavvenkatadri@ieee.org

Sayali Kulkarni
COE - CNDS
Veermata Jijabai Technological Institute
Mumbai
sayalikulkarni79@gmail.com

Vedant Athavale
COE - CNDS
Veermata Jijabai Technological Institute
Mumbai
vedantmilindathavale@gmail.com

Akhila Jagarlapudi
COE - CNDS
Veermata Jijabai Technological Institute
Mumbai
akhila.j@ieee.org

Shraddha Suratkar
COE - CNDS
Veermata Jijabai Technological Institute
Mumbai
sssuratkar@ce.vjti.ac.in

Faruk Kazi
COE - CNDS
Veermata Jijabai Technological Institute
Mumbai
fskazi@el.vjti.ac.in

Abstract— ***Practically, Deepfake technology has given people access to generate fake videos that look like real content using neural networks, and can further create misconceptions and deceit about the innocuous elements of society. This technology can prove fatal not only to national security but on an international level. Existing methodologies that apply deep learning to automatically extract salient and discriminative features to detect Deepfakes based on typical CNN-LSTM models tend to have their shortcomings. Having said that, we propose a system that extracts Spatio-Temporal features and achieves Real-Time Deepfake detection using Transformers. For the end user, a web application was developed, which with utmost simplicity allows the uploading of a video that will be further authenticated within the application and, at the same time, features the authentication of live meetings.***

Keywords— Image Processing, Deep Learning, Vision Transformer, Video Vision Transformer

I. Introduction

Deep learning increases the potential of artificial neural networks (ANN), which are designed to mimic brain activity. With the aid of deep learning, machines can be trained with a large amount of data and registering power. A deep learning application called Deepfake creates fake images, sounds, and recordings. Deepfakes uses a machine learning framework such as Generative Adversarial Networks (GAN) and Variational Auto Encoders (VAE).

Recent prominent occurrences of Deepfake videos have targeted former U.S. President Barack Obama [1], Facebook CEO Mark Zuckerberg [2], and the current president of Ukraine Zelensky Volodymyr [3] tampering with their public image and further fabricating fallacy. Traditional visual impacts or PC graphics approaches can still produce a significant portion of Deepfakes. These models analyze a person's facial articulations and advancements and combine them with facial photographs of other people. Deepfake techniques often need many images and videos to train models to produce photo-realistic images and recordings. Since there are so many recordings and images available online, well-known politicians and celebrities have been popular targets for the Deepfakes conundrum.

Because there is so much Artificial Intelligence (AI) based technology available online that is either free or cheap, producing manipulated or fraudulent films is now easier than ever. If a person or organization decides to employ Deepfake technology with bad intentions, it can seriously endanger the general populace due to how easily it can be used. Hence, it gives rise to the ever-demanding requirement to tackle this sensitive issue. Our proposed model attempts to placate the effects of Deepfakes which is computationally efficacious and enables Deepfake detection in real-time.

II. Related Work

According to Kurniawan Nur Ramadhani and Rinaldi Munir in [4], there are various methods for detecting deep fakes, including visual feature-based, local feature-based, deep feature-based, and temporal feature-based methods.

Deep learning techniques have been used successfully for digital picture forensics during the past few years. Barni et al. [5] used deep learning methods to locally identify double JPEG compression in photos. A network is suggested by Rao and Ni [6] to identify image splicing. Targeted by Bayar and Stamm [7] is any image general falsification. Computer graphics are distinguished from photographic images by Rahmouni et al. [8].

The approach for detecting face tampering in videos is presented by D. Afchar et al. [9] with an emphasis on Deepfake and Face2Face, two contemporary methods for producing fake videos that look incredibly realistic. Hence, to concentrate on the mesoscopic characteristics of images, this research uses a deep learning approach and provides two networks, each with a limited number of layers. MesoNet is a neural network developed specifically to recognize Deepfakes. MesoNet relies on a middle approach using a deep neural network because social media videos, like those on Instagram, are typically low-quality compressed videos, making microscopic analysis based on image noise impossible.

A different visual feature method proposed by Yang et al. [10] uses erratic head orientations. This technique evaluates the discrepancy in posture between the neck, shoulders, and facial muscles. To assess consistency, stance directions based on 68 facial landmarks and 17 facial landmarks, which reflected the pose direction from the middle area of the face, were compared.

According to Hady A. Khalil and Shady A. Maged in [11], deepfake detection in the case of face-swapping deepfakes, they are hard to detect when the person is facing straight towards the camera, when the person looks to his side, imperfections in the deepfake generated image can be seen. In [12], J C Dheeraj et al. provide a model using the Convolutional Neural Network (CNN) architecture that pre-processes the pictures using error level analysis (ELA).

Having a look at the pre-existing models above, we noticed that very few of the researchers were keen on the implementation of attention-based mechanisms. Of the very few methods utilizing attention mechanism, they had either used spatial attention or temporal attention but not both at the same time.

So, looking at this as an opportunity to exploit this research gap, we implemented a Video Vision Transformer which is an attention-based transformer model. Thus, our paper summarizes our research in the upcoming sections starting with Datasets and pre-processing, followed by our proposed model architecture and obtained results.

III. Dataset

The FaceForensics++ [13], employed for our work, consists of data which is based on DeepFakes [14], FaceSwap [15], Face2Face [16] and NeuralTextures [17], which are well-known examples of facial modifications at arbitrary compression levels and sizes. The dataset uses four automated, cutting-edge face manipulation techniques that are applied to 1,000 pristine videos that have been downloaded from the Internet and the accumulated videos are particularly from YouTube, to replicate realistic scenarios.

IV. Preprocessing

We analyze the data set in such a manner that we only pass the portion of the video in which the face is present and discard the other portions of the video to make acceptable predictions about the videos. There are numerous ways to accomplish this. We could do this by either detecting the face in every frame of the video or by detecting the face in the first frame and tracking it throughout the rest of the video. Using a portion of the dataset, we evaluated the two approaches listed below, and we chose the one that pre-processed the complete dataset the most quickly.

A. *Haar Cascades*

Detection of objects with the help of the Haar cascade is an effective method proposed by Paul Viola and Michael Jones [18]. This method, known as the "Rapid Object Detection using Boosted Cascade of Simple Feature Cascade" was proposed in 2001. This is a machine-learning-based method used to find objects in different images after the cascade function is trained with a set of positive and negative images. Using the above Haar-based features, we extracted the faces from a total of 9537 frames across 20 videos and tabulated the results for the same in Table I.

TABLE I.

TIME BASED RESULTS FOR HAAR CASCADE IMPLEMENTATIONS

Total Frames	**Total Time Required**	**Time per Frame**	**Success Rate**
9537	142.0614s	0.0149s	99.95%

Since the faces are extracted from individual frames in the video, the dimensions of the bounding box vary for every frame thus leading to a variable-sized input to the transformer model. Moreover, we are not utilizing the correlation between the adjacent frames while extracting the face from the video which poses a major drawback for this method. So, to solve this issue, we use Haar Cascades just to detect the face from the initial frames and then track that face for the entire span of the video. This helps us to have a fixed-size bounding box for every frame of the video. Thus, after detecting the face, we implement an object-tracking mechanism for face extraction and the methods that we use for face extraction are listed below.

B. *Object Trackers*

Trackers utilize all the information they have up to the point to track an object which is way more efficient, unlike detection algorithms like Haar Cascades, which start from scratch for every frame of the video. So, we implemented various trackers and performed a comparative analysis to select the best one. We start by detecting the face from initial frames in every video using Haar Cascades, followed by feeding the same into the object tracker which then tracks the object in all the subsequent frames of the video. This method is more effective because of the utilization of the correlation between the adjacent frames in the video. The results obtained through it are shown in Table II.

TABLE II.

TIME BASED COMPARISON OF VARIOUS TRACKERS

Tracker	**Total Frames**	**Total Time Required**	**Time Per Frame**	**Success Rate**	**Bounded/ Unbounded**
KCF [19]	9537	678.67s	0.0712s	94.35%	Bounded
CSRT [20]	**9537**	**527.38s**	**0.0553s**	**97.64%**	**Bounded**
MIL [21]	9537	657.40s	0.0689s	97.64%	Bounded
TLD [22]	9537	631.13s	0.0662s	97.43%	Unbounded
MedianFlow [23]	9537	77.31s	0.0081s	97.46%	Unbounded
MOSSE [24]	9537	53.14s	0.0056s	96.25%	Unbounded

After performing a thorough analysis on various trackers and obtaining the results we see that CSRT is the preferred one for preprocessing the data for many reasons. First of all, it is a bounded tracker which means that the tracker's output has fixed dimensions contrary to the unbounded ones. Generally, for longer videos, unbounded trackers tend to be unstable and the dimensions of the output tend to increase or decrease monotonically, and resizing the frames leads to the loss of information. Although in some scenarios unbounded trackers can be used if the proximity of the subject varies widely and is generally helpful when the subject moves closer to the camera over time. But in scenarios when the subject is moving away, the model might fail to capture the essence of the motion of the subject leading to poor performance. But in such scenarios, the model performs much better on the data preprocessed by bounded trackers since the entire motion is captured in the same frame size. Another advantage of the CSRT tracker is that among the bounded trackers it is the fastest. Hence, we choose CSRT for preprocessing the data.

This algorithm [20] adjusts the filter support to the part of the selected region from the frame for tracking using spatial reliability maps, allowing it to expand the search area and track non-rectangular objects. Reliability indices are weights for localization that reflect the quality of the studied filters by channel. As a result, when Histogram of Oriented Gradients (HoG) and Colornames are used as feature sets, the algorithm performs reasonably well.

C. *Video Segmentation and Grouping*

With a large dataset to train, data pre-processing requires a lot of time, making this aspect of model training a bottleneck. Transformers generally perform better than Long Short-Term Memory (LSTM) Networks as they have a fixed input size rather than a variable-sized input. Hence to have a fixed-size, every data point was formed by grouping a series of 8 continuous frames of a particular video. This enables us to capture the spatial as well as temporal features. Thus, instead of treating the entire video as a single data point every video is treated as a group of several data points and predictions are made on this group of data points and a cumulative prediction for the entire video is gained by averaging over all of those data points for that particular video. This 8-frame video represents one training sample.

After analyzing the data, we found out that the average frames in each video were 510. So, there will be 510 divided by 8 (which is the group of frames we are considering at once) is approximately equal to 64 training examples generated from each video. Instead of saving these videos separately, we create a JSON file in which each entity represents the individual training sample, and corresponding to it is the path of the video it is part of and the range of frames that are present in that particular sample. This is further shuffled and the resultant JSON file is fed to the Data Generator which extracts the training sample from the original dataset depending on the range of frames specified in the JSON file.

V. The Proposed System

Similar to ViT: Vision Transformer [25], ViViT (Video Vision Transformer) [26] uses a transformer encoder, and a position and token Embedding mechanism. The sole distinction is tokenizing many frames at once and allowing the transformer to pick up on spatial and temporal connections. A sequence is made up of tokens.

Since the transformer model can only accept vectors as input, every token should correspond to a vector; these vectors are referred to as embedding vectors. Two straightforward techniques, Tubelet Embedding and Uniform Frame Sampling, were taken into consideration while mapping a movie to a list of tokens. However, Uniform Frame Sampling only accounts for spatial features. Hence, we consider Tubelet Embedding to incorporate both spatial and temporal features. This is enhanced by positional embeddings, and the Transformer is then given this data as input. After extracting non-overlapping, Spatio-temporal tubes from the input video clip, tubelet embedding linearly projects the tubes (Fig. 1). The Spatio-temporal data is combined using the tubelet embedding approach throughout the tokenization phase.

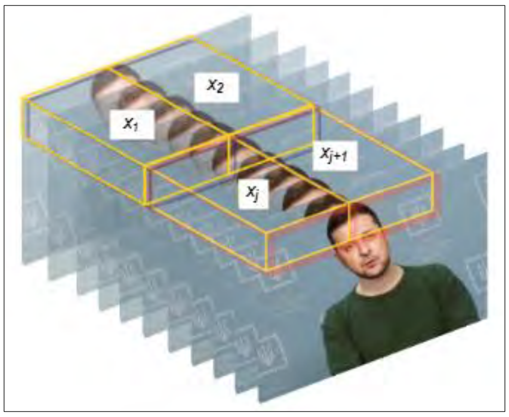
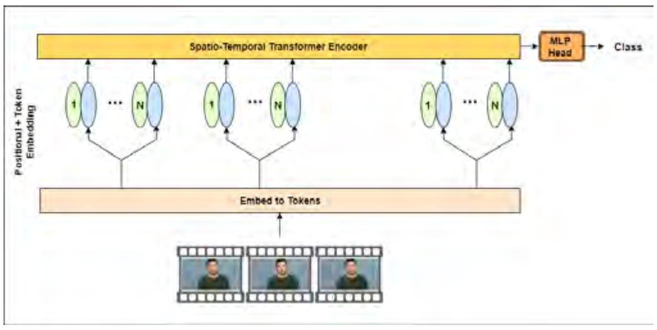


Fig. 1. Tubelet Embedding

When it comes to transformer models for video, in [26], multiple transformer-based architectures have been proposed. They began with a straightforward extension of ViT [25] that models pairwise interactions between all Spatio-temporal tokens, followed by the development of more efficient variants which factorize the spatial and temporal dimensions of the input video at various levels of the transformer architecture.

A. *Spatio Temporal Attention*

In this method, the transformer encoder is used to deliver all spatio temporal tokens extracted from the video. In contrast to Convolutional Neural Network (CNN), every transformer layer depicts all pairwise interactions between spatio temporal tokens. Now that the first token of the first image may interact with the last token of the last image, long-distance interactions between the videos from the first layer are feasible. The second model overcomes the quadratic complexity that the first model has as a result [26].



Spatio-Temporal Transformer Encoder
MLP Head
Class
Positional + Token Embedding
1
...
N
Embed to Tokens

Fig. 2. Spatio-Temporal Attention

B. *Factorised Encoder*

This paradigm factorizes the encoder, as the name implies, which means that all frames are first passed through a spatial encoder, which means that each frame is sent through a different spatial encoder. The temporal encoder receives the output features from the spatial encoder and processes them. This makes late temporal information fusion easier since spatial information is obtained first, and subsequently temporal information is fused. [26]

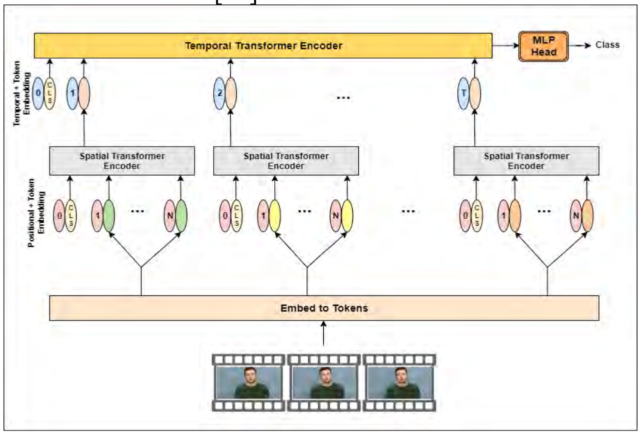

Temporal Transformer Encoder
MLP Head
Class
Temporal + Token Embedding
Spatial Transformer Encoder
Spatial Transformer Encoder
Spatial Transformer Encoder
Positional + Token Embedding
Embed to Tokens

Fig. 3. Factorised Encoder

Because there are two encoders instead of one, this model [26] has more parameters, but it has fewer floating-point operations because nothing interacts with everything directly. Although the tokens communicate with each other across great distances, they do so in an indirect manner, making this model significantly simpler than Fig. 2.

C. *Factorised Self Attention*

The spatial and temporal self-attention encoders, which are used in [26] are blocks within the single encoder that they used in their model. This model is more complicated than the Factorised Encoder Model (Fig. 3) because nothing directly interacts with anything else, although it only has one encoder and the same number of transformer layers as the Spatio-temporal Attention Model. Before conducting spatial attention, it reshapes the input tokens as several frames into the spatial dimension. Then it reshapes it suitably so that it may be fed into the temporal self-attention block, where it performs temporal self-attention and passes the result to the next transformer block. Overall, experiments were conducted to determine which block should be placed first in the arrangement of the blocks. They investigated whether the spatial or temporal blocks should be used first, and discovered that it makes no difference.

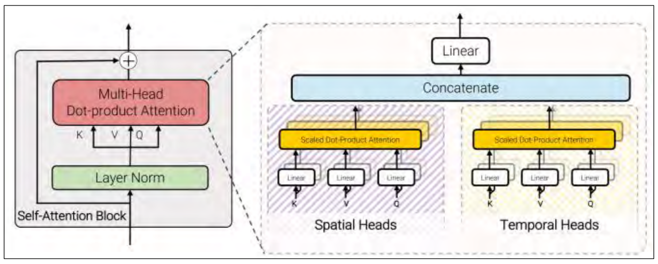

Multi-Head Dot-product Attention
K
V
Q
Layer Norm
Self-Attention Block
Linear
Concatenate
Scaled Dot-Product Attention
Scaled Dot-Product Attention
Linear
Spatial Heads
Temporal Heads

Fig. 4. Factorized Dot-product Attention [26]

D. *Factorised Dot-Product Attention*

In [26], a model with the same number of parameters as Fig. 2, but the same computational complexity as Fig. 3 and 4 has been proposed. This strategy is a little trickier because it divides everyone's attention into two parts before anything else. Because dot-product attention is computed over only the spatial axis for half of the heads and only the temporal axis for the other half, and then keys are created for the attention heads, the concept of neighbourhood is essential in this context. Changing the keys and values in each query such that it only examines tokens from the same spatial and temporal index is the basic principle. By computing Ys = Attention(Q, Ks, Vs) for half of the attention heads and Yt = Attention(Q, Ks, Vs) for the remaining attention heads, the model then attends to tokens from the same spatial dimension (Q, Kt, Vt). Finally, a linear projection is employed to integrate the outputs of several heads once they have been concatenated. The Attention(Q, K, V) is given:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d}})V$$

Where,

K, Q and V are Key, Query and Value vectors respectively. and, D is vector dimensionality.

VI. The Proposed System Architecture

Our model uses the ViViT (Video Vision Transformer) architecture which is based on the Vision Transformer Architecture except for the fact that the model has been modified to take several consecutive video frames instead of images.

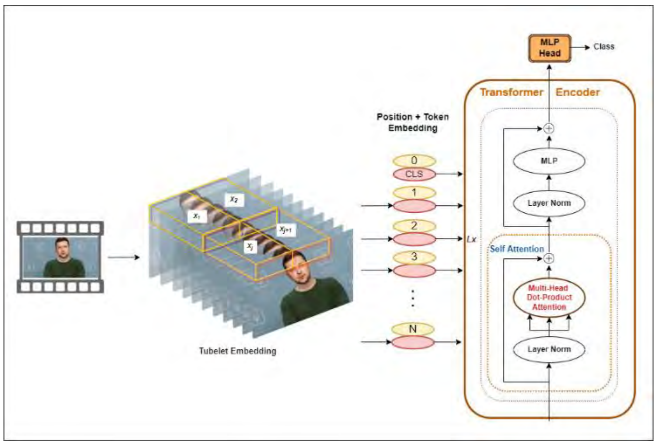


Fig. 5. Video Vision Transformer with Spatio-Temporal Architecture

The output of the positional encoder is passed to the first transformer block. The transformer block consists of a batch normalization layer which normalizes the entire batch of data. This is passed to the Multi-Headed Attention (MHA) layer which is the most important layer of a transformer block. The MHA layer is responsible for providing the attention mechanism for the detection process. The output of the Multi-Headed Attention (MHA) layer is the attention weights that add to the input of that particular block.

Then the output is passed to the two-layer feed-forward network with a skip (residual) connection. The output is then normalized and passed to the next transformer block. This procedure is repeated for 8 transformer blocks.

VII. Results

The training, testing, and validation accuracy metrics are shown below in Table III. This section provides a summary of how our model compares to other models. We evaluated several models' training and testing accuracy, as well as their loss, to ours. The inference time of several models is also shown in Table V. It demonstrates that our model has higher accuracy and lower loss. Another noticeable difference is that our model (Fig. 5) requires substantially less time to analyze frames, resulting in a shorter inference time. Parallel processing enabled by Transformer significantly improves the model's frame rate.

Our CPU Specification is Dual 20-Core Intel Xeon E5-2698 v4 2.2 GHz. Our GPU Specification is NVIDIA GTX 1650 and our Graphic Memory is 4GB. The predictions were then retrieved from our data, which were divided into two inference methods, the first into groups of frames and the second into videos. All of the videos are broken into little parts of 8 frames. Each chunk is considered an individual training and testing sample. Alternatively, while viewing a complete video, predictions are produced in smaller chunks of 8 frames each, which are then compiled to make appropriate predictions on the entire video and classify them as real or fake. As a consequence, even if one portion produces an incorrect result, other sections compensate for it. As a result, the overall accuracy of prediction on the video is improved. Table IV contains the four measures and their associated values. These measures include F1-score, precision, recall, and so on. [27]

TABLE III.

MODEL'S PERFORMANCE EVALUATED BY VARIOUS METRICS
(AS A GROUP OF 8 FRAMES)

Score	Train	Test	Val
Accuracy	95.805%	93.127%	93.462%
F1 – Score	0.952	0.924	0.919
Recall	0.935	0.896	0.886
Precision	0.971	0.953	0.954

TABLE IV.

MODEL'S PERFORMANCE EVALUATED BY VARIOUS METRICS
(AS VIDEOS)

Score	Train	Test	Val
Accuracy	99.500%	98.250%	98.250%
F1 – Score	0.995	0.982	0.972
Recall	0.990	0.975	0.970
Precision	0.100	0.990	0.995

Even if there are glitches in some frames of the video, the overall performance of the model for the entire video would be much better and the model would be resilient to glitches. This can be seen while comparing Table III with Table IV.

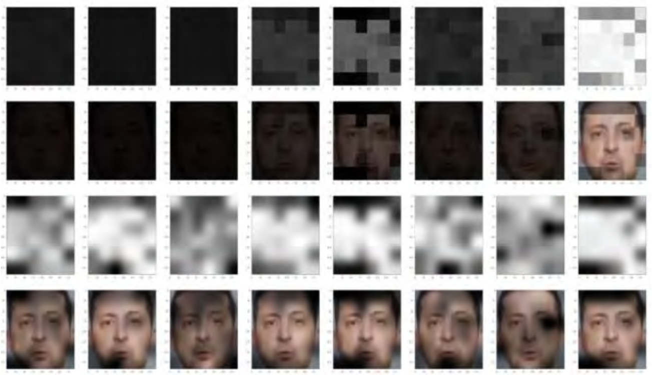


Fig. 6. Visualization of Attention Filters

The attention map allows us to see where the model is concentrating at any given instant. Some of the frames from the video Zelensky Volodymyr's DeepFake are shown here (Fig. 6). For each transformer block that is present in the model, an attention map may be seen. Each Tubelet portion constitutes a component of a 6 by 6 matrix in each Attention map. For visualization, we smoothened the Attention Maps using Gaussian Filters.

VIII. CONCLUSION

In this research, we have presented a Deep Fake detection system that uses Video Vision Transformer (ViViT) to expose any video in real time. The model reaches a high level of accuracy and produces reliable results in real time. We created a web application to make the model accessible to the end user where the user has two options for video authentication, basic video authentication and live meeting authentication.

Basic video authentication allows the user to check the authenticity of any available video. Moreover, if the user is in a meeting and wishes to check the authenticity of the video stream of the person in the meeting, they may choose the option of authenticating a live meeting, which allows them to record a segment of the live meeting in which they are present. Using the recorded segment, the model authenticates the video and returns the predictions. (Fig. 7) is a screenshot of the web application. The above feature allows the user to fully utilize the model's real-time application: examine the meeting while being present in it. The potential for applications built on deep learning and computer vision techniques is nearly endless.

TABLE V.

OUR MODEL'S ACCURACY VS ACCURACY FOR DIFFERENT MODELS

Network Name	Parameters	Training		Testing		Inference Time	
		Accuracy	Loss	Accuracy	Loss	CPU	GPU
Mesonet [9]	28K	73.189%	0.25	72.39%	0.2392	194 ms	180.7 ms
ResNet-50 [28]	25.6M	75.26%	0.0655	74.12%	0.1505	1978 ms	1142.2 ms
VGG-19 [29]	143.7M	74.92%	0.010	73.28%	0.033	302.2 ms	254.3 ms
Xception [30]	22.9M	77.83%	0.117	75.99%	0.16	1080 ms	1002 ms
ViViT-DF-96M (Ours)	**96.6M**	**96.17%**	**0.05**	**92.36%**	**0.1172**	**3.846 ms**	**2.5252 ms**

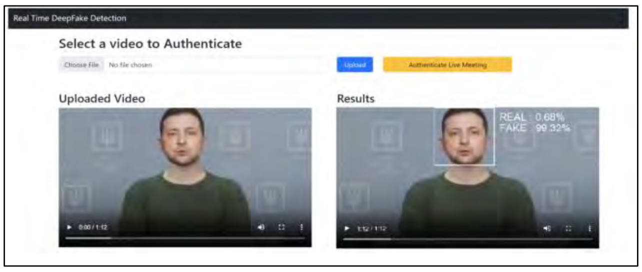


Fig. 7. Application to authenticate a real-time video

In the future, we intend to combine audio and video, which will allow the model to include more features. Additionally, we intend to do quantization, which will speed up the model and decrease the inference time. As previously said, our model's attention maps describe (using explainable AI) the areas that the model emphasizes on.

Further, the inference time may significantly decrease when we consider these attention maps to focus only on certain sections of the image rather than the entire image. However, accuracy may suffer as a result, therefore there is a trade-off between these two factors when simply employing attention maps.

IX. References

[1] "Fake Obama created using AI video tool - BBC News," YouTube, uploaded by BBC News, 19 July 2017. [Online]. Available: https://www.youtube.com/watch?v=AmUC4m6w1wo.

[2] "Artists create Zuckerberg 'deepfake' video," YouTube, uploaded by TimesLive Video, [Online]. Available: https://www.youtube.com/watch?v=cnUd0TpuoXI.

[3] "Deepfake video of Volodymyr Zelensky surrendering surfaces on social media," YouTube, uploaded by The Telegraph, 17 March 2022. [Online]. Available: https://www.youtube.com/watch?v=X17yrEV5sl4.

[4] K. N. Ramadhani and R. Munir, "A Comparative Study of Deepfake Video Detection Method," in *2020 3rd International Conference on Information and Communications Technology (ICOIACT)*, 2020.

[5] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo and M. Maggini, "Aligned and nonaligned double jpeg detection using convolutional neural networks," *Journal of Visual Communication and Image Representation,* vol. 49, pp. 153-163, 2017.

[6] Y. Rao and J. Ni, "A deep learning approach to detection of splicing and copy-move forgeries in images," in *Information Forensics and Security (WIFS) 2016 IEEE International Workshop*, 2016.

[7] B. Bayar and M. C. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, 2016.

[8] N. Rahmouni, V. Nozick, J. Yamagishi and I. Echizen, "Distinguishing computer graphics from natural images using convolution neural networks," in *IEEE Workshop on Information Forensics and Security WIFS 2017*, December 2017.

[9] D. Afchar, V. Nozick, J. Yamagishi and I. Echizen, "MesoNet: a Compact Facial Video Forgery Detection Network," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018.

[10] X. Yang, Y. Li and S. Lyu, "Exposing Deep Fakes Using Inconsistent Head Poses," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[11] H. A. Khalil and S. A. Maged, "Deepfakes Creation and Detection Using Deep Learning," in *2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, Cairo, Egypt, 2021.

[12] J. C. Dheeraj, K. Nandakumar, A. V. Aditya, B. S. Chethan and G. C. R. Kartheek, "Detecting Deepfakes Using Deep Learning," in *2021 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, 2021.

[13] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," in *arXiv:1901.08971v1 [cs.CV]*, 2019.

[14] "https://github.com/deepfakes/faceswap, Deepfakes github.," Accessed: 2018-10-29.

[15] "https://github.com/MarekKowalski/FaceSwap/, Faceswap.," Accessed: 2018-10-29.

[16] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt and M. Nießner, "Face2Face: Real-Time Face Capture and Reenactment of RGB Videos," *in IEEE Conference on Computer Vision and Pattern Recognition,* p. 2387–2395, June 2016.

[17] J. Thies, M. Zollhofer and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," *ACM Transactions on Graphics 2019 (TOG),* 2019.

[18] OpenCV, "Face Detection using Haar Cascades," [Online]. Available: https://tinyurl.com/2p96sxnz.

[19] J. F. Henriques, R. Caseiro, P. Martins and J. Batista, "High-Speed Tracking with Kernelized Correlation Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 37, no. 3, pp. 583 - 596, 2014.

[20] A. Lukežič, T. Vojíř, L. Č. Zajc, J. Matas and M. Kristan, "Discriminative Correlation Filter Tracker with Channel and Spatial Reliability," *International Journal of Computer Vision,* vol. 126, pp. 671-688, 2019.

[21] B. Babenko, Ming-HsuanYang and S. Belongie, "Visual tracking with online Multiple Instance Learning," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, 2009.

[22] Z. Kalal, K. Mikolajczyk and J. Matas, "Tracking-Learning-Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 34, no. 7, pp. 1409 - 1422, 2011.

[23] Z. Kalal, K. Mikolajczyk and J. Matas, "Forward-Backward Error: Automatic Detection of Tracking Failures," in *International Conference on Pattern Recognition*, Istanbul, Turkey, 2010.

[24] D. S. Bolme, J. R. Beveridge, B. A. Draper and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, 2010.

[25] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit and N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," in *ICLR*, 2021.

[26] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić and C. Schmid, "ViViT: A Video Vision Transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[27] S. Pashine, S. Mandiya, P. Gupta and R. Sheikh, "Deep Fake Detection: Survey of Facial Manipulation Detection Solutions," in *International Research Journal of Engineering and Technology* , 2021.

[28] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," in *arXiv:1603.05027v3 [cs.CV]*, 2015.

[29] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *arXiv:1409.1556v6 [cs.CV]*, 2014.

[30] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in *arXiv:1610.02357v2 [cs.CV]* , 2016.

[31] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," in *arXiv:1901.08971 [cs.CV]*, 2019.