Fake video detection

Internship report by Nicolai Popov

Contents

1 Introduction

This research work is focused on verification of the face videos against «deepfake attacks». Such type of verification is a part of a broader domain called liveness detection. Its meaning is defined at web resource as [49]: . . . computer's ability to determine that it is interfacing with a physically present human being and not a spam bot, inanimate spoof artifact or injected video/data». There are five types of different attacks, described at the same website [49] that form two groups. First group includes physical and digital presentation attacks. It means some physical artifacts, such as printed photos, rubber masks or digital photos or videos displayed on a screen are presented to the camera. The second group comprises bypass attacks, such as digital replay attacks and attacks where the video frames are altered to take over the camera feed itself. So-called deepfake attacks belong to the second group and we take into account liveness verification against only this type of attacks.

So-called «deepfake» is not an exact term. Wikipedia defines it as «synthetic media in which a person in an existing image or video is replaced with someone else likeness.» Nowadays, it is used to define results of different techniques of fake media generation such as face swap, reenactment, synthesis of portraits of non-existent people, portrait attribute editing and portraits morphing attack. Face swap method swaps the face from source to target, but preserves face expression and head pose of the target. Reenactment technique transfers face expression and head pose from source to target, but keeps face of target, i.e. identity, intact. Method of non-existent portrait generation does not use source and target portraits and directly creates artificial examples. Portrait attribute editing technique uses only a source portrait and changes such features as the color of eyes and hair, length of the hair, gender and age of the person and so on. Portraits morphing is used to join portraits of several persons into one portrait that can be used to bypass face recognition systems for both persons using only one synthesized face image. So the deepfake term in this report will stand for fake visual media.

Liveness verification pipeline created at ShareID company comprises the following steps. A user is told to perform several actions in front of a smartphone camera: smiling, blinking, turning his/her head left and right. Successful execution of these actions is a part of liveness test pass. Therefore, only face swap and reenactment techniques may be used to generate fake video of a target person performing abovementioned actions. These fake generation methods process media of the two persons, i.e. images and videos, that are called source and target. Target image or target identity are the ones with face, expression or head pose being changed. Source image or identity are the ones that provide the face (identity) or face expression and head pose for transfer to the target.

There are lots of papers published and research done in order to develop algorithms that can determine that a video under examination is pristine or manipulated one. From the other side, great amount of papers is dedicated to fake media synthesis. The aim of this work is to improve the quality scores of a SOTA model which could be used for the detection of face swap and reenactment techniques.

The first two deepfake generation examples appeared in 2016 and 2017. From that point the amount of published research articles related to the «deepfake» topic rises exponentially starting from zero papers in 2017 and exceeding hundreds of papers in the last year. This proves that the problem of fake media detection is important and really demanded. Therefore, technologies that can automatically assess the integrity of digital visual media are indispensable. Information became the most precious resource and the most powerful instrument in the age of Internet. The speed of some post going viral in social networks is raising concerns in view of availability of deepfake generation technology. Danger of using deepfakes can be divided into four main categories. Societal aspect includes increasing social unrest and political polarization. The example for legal category is falsifying electronic evidence. Next one, probably the most wide-spread, is personal threat, i.e. harassment and bullying, non-consensual adult videos and online child exploitation. Of course, threat for traditional cybersecurity is also present and includes extortion, fraud and manipulating financial markets. All these threats can be solved by fake detection algorithms.

The research made on the topic of deepfake can be divided into two main directions, where the first group improves quality of generated fakes and the second one tries to make stronger detection algorithms. But it seems, that deepfake production technology is currently improving faster than deepfake detection technology. This research is dedicated to making a contribution to this constant competition between generation and detection methods to support the latter.

2 Related work

The following review of relevant articles is grouped into several parts. Generation methods that perform either reenactment or face swap are described first. Each of these two types of generation is also categorized taking into account the technology used under the hood which is either learning algorithms based on data or computer graphics. Second part comprises the detection methods that are applicable to video classification with no sound. This condition is imposed by the application pipeline developed at ShareID company.

2.1 Face swap for fake media generation

Face swap method allows to transfer the face (i.e. identity) of the person from source image or video to the person in the target image or video while preserving head pose and face expression of the latter intact.

2.1.1 Learning based methods

Name «deepfake» originates from the first public deepfake synthesis model [50] that actually implements a learning based face swap algorithm. As described in paper [10], to generate a Deep Fake, the algorithm is fed with an image that contains the source face to be replaced. Bounding boxes of faces are obtained with a face detector, followed by the detection of facial landmarks. Next step is face alignment, when face area is transformed to a standard configuration. To perform this an affine transformation is found by minimizing the alignment errors between central facial landmarks and a set of standard landmark locations. This warped image is then fed into the deep generative neural network to create the synthesized face. The latter is transformed back to match the original face. Finally, after boundary smoothing a Deep Fake image is created. Models used for deep fake generation are two auto-encoders with an encoder shared between source and target images. The encoder-decoder models are trained for the source and the target faces to reconstruct these images. To create a fake image, the shared encoder is used before the decoder of the source face and this model is applied to the target face to transfer its identity to the source face. For each pair of persons one such auto-encoder model is trained. Faceswap-GAN is a similar algorithm presented in [10] which uses CycleGAN in place of an encoder-decoder network. The VGGFace perceptual loss is added to make eye movements to be more realistic. The CycleGAN with adversarial loss is utilized for generative network in this model.

Few-Shot Face Translation model [51] is capable of producing faces with gaze direction, glasses, and hair occlusions being consistent with the given source face. But it has suboptimal performance in terms of translating to asian faces as reported in the paper. In this approach generator is an encoder-decoder based network too. The swap-them-all approach is basically a GAN conditioned on the latent embeddings extracted from a pre-trained face recognition model. Other objectives that aimed to improve translation performance while keeping semantic consistency, such as perceptual loss on RGB output and cosine similarity loss on latent embeddings, are also introduced.

DeepFaceLab [11] application is claimed to be the best open source application. Its pipeline immcludes the following steps. First, face detection is performed with a face detector, for example RetinaFace. Next, face alignment is performed using point pattern mapping and transformation method to calculate a similarity transformation matrix. Face segmentation is done with deep nets generating fine-grained masks. Pipeline consists of shared encoder, two decoders for source and target and performs face-swapping task. The generated face is transformed with its mask from target decoder to the original position of the target image in source image. Than Poisson blending and sharpening are performed. Finally, a pre-trained face super-resolution neural network is used to sharpen the blended face.

Dfaker model [52] inputs an image and outputs a pair of images: one RGB with the reconstructed face, and a grayscale one to be used as a mask to guide which sections of the image are to be replaced. For the reconstructed face the masked DSSIM loss is used. It behaves as a standard SSIM difference measure in the central face area and always returns zero loss in the surrounding background area outside of the face so as not to train irrelevant features. MSE loss is used for the mask.

FaceShifter [15] is a generation model well-known in the community. The method requires two input images, source image and target image to provide attributes, e.g. pose, expression, scene lighting and background. The swapped face image is generated through a two-stage framework. In the first stage, a NN-1 generates a high fidelity face swapping result. In the second stage, another NN-2 handles the facial occlusions and refines the result. NN-1 consists of identity encoder, multi-level attributes encoder and adaptive attentional denormalization generator. Pretrained state-of-the-art face recognition model is used as identity encoder. The identity embedding is defined to be the last feature vector generated before the final FC layer. Authors believe that this model can provide more representative identity embeddings than the 3D-based models like 3DMM. The target image is fed into a UNet-like structure. The embeddings for face attributes, such as pose, expression, lighting and background, are represented by feature maps generated from the UNet decoder. Adaptive Attentional Denormalization layer is an alternative to simple concatenation of identity and attributes feature maps. It was proposed in this paper to tackle the issue of blurry swapped images. It consists of batch normalization of activation map followed by separate identity and attributes integration and finally weighted with an attention mask. The training loss has three components: one for keeping the identity the same in the source and output, second for keeping the attributes the same in the target and output and third for having the output the same as the target if target and source are the same image. NN-2 is used to delete occlusions that appear on the target face in swapped image. Refined image is obtained in 2 steps.

2.1.2 Graphics based methods

FaceSwap [53] is a graphics-based approach. First, the face region is detected in the source image and its landmarks are found. Then, 3D model is fitted to these landmarks. The vertices of this model are projected to the image space and are used as texture coordinates. For each video frame the following steps are taken: face region is detected and the facial landmarks are located, 3D model is fitted to the located landmarks, the 3D model is rendered with the textures calculated before from the source face. The image of the rendered model is blended with the image obtained from the camera using alpha blending and color correction.

Face Swapping algorithm [21] swaps a face from a source image to a target image treating both images the same, apart from the final stage. Method first localizes 2D facial landmarks in each image using an off-the-shelf detector for this purpose. Using these landmarks, viewpoint is computed and the 3D shape is modified to account for face expression. Then, faces are segmented from their backgrounds and occlusions using a FCN trained to predict per-pixel face visibility. Finally, the source is efficiently warped onto the target using the two, aligned 3D face shapes as proxies, and blended onto the target image.

2.2 Face reenactment for fake media generation

Reenactment method transfers face expression and head pose of the person in source media to the person in the target media while keeping the identity of the latter the same.

2.2.1 Learning based methods

Neural Textures method was presented in paper [8] as a neural textures-based rendering approach among other applications of the technique. Method uses the original video data to learn a neural texture of the target person, including a rendering network. Deferred Neural Renderer is a encoder-decoder network with skip-connection, similar to U-Net, that forms a photo-realistic image given a screen space feature map. The parameters of Neural Renderer and neural textures are learned. Neural Textures is a feature map that stores a high-dimensional learned feature vector per texel. The texel-to-pixel mapping for each of the training images is pre-computed before training. Training pairs consist of this map and the corresponding ground truth color image. The $L1$ loss is defined on random crops of the full frame image. Steps of image synthesis are the following: neural textures are sampled based on the uv-map using the standard graphics pipeline resulting in a feature map in the target image space. The latter is the input to Neural Renderer which outputs the final image that photo-realistically re-synthesizes the original object. Using expression transfer, authors generate an altered uv-map of the target actor matching the expression of the source actor. This uv-map is used to sample from the neural texture of the target actor. In addition, they provide a background image to the neural renderer, to output the final reenactment result. For facial reenactment, a variety of facial expressions in the training video is necessary, otherwise it loses expressiveness. In the experiments with transferring the expression of the Obama to the face of Macron, authors used 650 training images for president Macron and 2400 for president Obama.

Deep Video Portraits technique [9] transfers the full 3D head position, head rotation, face expression, eye gaze, and eye blinking from a source actor to a portrait video of a target actor, while preserving the target's identity and appearance. The core of the approach is a generative neural network with a space-time architecture. The network takes as input synthetic renderings of a parametric face model, based on which it predicts photo-realistic video frames for a given target actor. The pipeline consists of the following steps. First, algorithm tracks the source and target actor using a state-of-the-art monocular face reconstruction approach that uses a parametric face and illumination model. The resulting sequence of low dimensional parameter vectors represents the actor's identity, head pose, expression, eye gaze, and the scene lighting for every video frame. This allows to transfer the head pose, expression, and/or eye gaze parameters from the source to the target, as desired to generate new synthetic renderings of the target actor based on the modified parameters. In addition to normal color rendering, correspondence maps and eye gaze images are also rendered. These renderings serve as conditioning input to the rendering-to-video translation network which is trained to convert the synthetic input into photo-realistic output.

Marionette [12] is a few-shot face reenactment method. A conditional generator generates the reenacted face given the source and K target images, and the discriminator predicts whether the image is real or not. Generator components are the following: preprocessor utilizes a 3D landmark detector to extract facial keypoints and yields landmark image, source encoder extracts pose and expression information from the source input and produces driver feature map. Target encoder adopts a U-Net architecture to extract style information from the target input and generates target feature map along with the warped target feature maps. Next, blender network receives driver feature map and target feature maps to produce mixed feature map. This is afterwards used by decoder as well as warped target feature maps to synthesize reenacted image. The decoder improves the quality of reenacted image exploiting proposed target feature alignment. Image attention block is proposed to solve the problem of losing spatial information of targets when a spatial-agnostic vector is used to transfer style information of targets to the source.

Controllable Face Image Generation technique [13] Given a collection of real face images, the goal of the authors is to train a network that generates realistic face images from random noise. They consider latent variables for five independent factors, such as identity, expression, illumination, pose, and a random noise accounting for other properties such as background. As in standard GAN, a discriminator is applied to compete with generator. This method can be used to embed real images into the latent space and edit their features in a disentangled manner, i.e. change expression, head pose, lighting. Projection to latent space would enable one to use this method to reenact a source portrait. To learn how a face image should be generated following the desired properties an imitation loss is used, but to fortify disentanglement, authors enforce the invariance of the latent representations for image generation using a contrastive loss. 3DMM model is used in both losses. The parameters of the 3D face such as identity, expression, reflectance, illumination and Euler angles for the head pose encode all information necessary to render a face. Objects from the space of this parameters are fed to generator, but first latent variable from W+ space is mapped to parameter space with learned VAEs, one per each of 5 modes. StyleGAN structure is used and, firstly, is trained with adversarial and imitative loss until seeing real images to obtain reasonable imitation. Then contrastive loss is added into the training. The editing of images is done in W+ space of StyleGAN generator by calculating the change of latent vector w+ corresponding to the change of the vector in parameter space: image is back projected to W+, a step is made in W+ and then new latent vector is fed into generator.

StyleRig method from [14] uses network RigNet which is trained between the 3DMM's semantic parameters and StyleGAN's input. Method generates portrait images with the photorealism of StyleGAN and provides explicit control over the 3D semantic parameters of the face. Parameter vector that encodes a face includes facial shape, skin reflectance, facial expression, scene illumination, head rotation and translation. Given a latent vector from W+ space that corresponds to an image, and a vector of wanted semantic control parameters, authors learn a function that outputs a modified latent code. The modified latent code should map to a modified face image that obeys the control parameter vector. So to reenact a real portrait a good backprojector for StyleGAN's W+ space is necessary. The pipeline includes a differentiable face reconstruction network (DFR) that maps the latent code to parameter vector. Two cycle-consistent losses are used to control the transferring of pose/expression/illumination from image v to latent vector. Consistency loss is the same but is used to control that the parameters that should not be changed by the performed edit operation, did not change. StyleRig allows control over the pose, expression, and illumination parameters of StyleGAN generated images.

First Order Motion described in [16] animates an object depicted in a source image based on the motion of a similar object in a driving video. Model is trained to reconstruct the training videos by combining a single frame and a learned latent representation of the motion in the video. It learns to encode motion as a combination of motion-specific keypoint displacements and local affine transformations. At test time the model is applied to pairs composed of the source image and each frame of the driving video and perform image animation of the source image. Framework is composed of two main modules: the motion estimation module and the image generation module. The purpose of the motion estimation module is to predict a dense motion field from a frame of the driving video to the source frame. The motion field is modeled by a function that maps each pixel location in driving video with its corresponding location in source image and is often referred to as backward optical flow. At inference, firstly, the locations of the key points are separately predicted for source at target media by an encoder-decoder network. The keypoint representation acts as a bottleneck resulting in a compact motion representation. For each keypoint the affine transformation is used to approximate local motion. A dense motion network combines the local approximations to obtain the resulting dense motion field and an occlusion mask that indicates which image parts of driving video can be reconstructed by warping of the source image and which parts should be impainted, i.e. inferred from the context.

X2Face model proposed in [17] has two inputs: a driving and a source frame. The source frame is fed to the embedding network and the driving frame to the driving network. The embedding network learns a bilinear sampler to determine how to map the source frame to a face representation, the embedded face, and the outputs is a 2-channel image of the same dimensions as the source frame that encodes the flow for each pixel. The driving network takes a driving frame as input and learns a bilinear sampler to transform pixels from the embedded face to produce the generated frame.

Talking head algorithm [18] assumes the availability of several video sequences, containing talking heads of different people. Face landmarks for all frames are used. There are three networks in the pipeline, an embedder that takes a video frame and an associated landmark image and maps these inputs into an N-dimensional vector. Next, generator takes the landmark image for the video frame not seen by the embedder, the predicted video embedding and outputs a synthesized video frame. It is trained to maximize the similarity between the outputs and the ground truth frames. The last net, the discriminator takes a video frame, an associated landmark image and predicts a realism score scalar value that indicates whether the input frame is a real frame of video sequence and whether it matches the input pose.

DaGAN model presented in the paper [20] consists of a generator and a discriminator. Given a source image and a frame from the driving video, authors exploit a pretrained depth estimation net to produce depth maps for each image. These depth maps and their RGB images are concatenated to learn geometry and appearance features for detecting face keypoints which can be used to generate relative motion fields of the human faces. The feature warping module accepts the keypoints as input to generate motion fields which are used to warp the source image feature map and fuse the motion with the appearance information, resulting in a warped feature map for the manipultad image.

2.2.2 Graphics based methods

Face2Face method [7] was the second public portrait manipulation method that appeared after the first «deepfake» method [50]. It is a graphics-based facial reenactment algorithm that transfers the expressions of a source video to a target video while maintaining the identity of the target person. For each frame in video a 3DMM head and face model combining identity, expression and reflectance blend-shapes. An energy function which is equal to weighted sum of photo-consistency, feature alignment and regularization terms is optimized to find weights for all parameters. Expression weights for neutral and current frame face for source face are used to calculate the deformation gradients with are the matrices for triangle deformations of the head mesh. These gradients and neutral target expression are used to optimize the expression vector for the current target frame which actually is the transferred expression. Alpha blending between the target video frame, projected mouth frame and the re-rendered face model with new expression results in the image with new expression and old identity.

Bringing Portraits to Life [19] is a face manipulation technique that uses a set of control points in the target image with neutral face expression to transfer face expression from a driving video that has a frame with neutral face too. The result is a video with a person from source image performing the same movements as person in source video. An aligning transformation is found between sets of control points that include facial landmarks. The corresponding pixel locations in the target image are warped using the aligning transformation found before. Facial landmarks together with the peripheral points, that do not move throughout the animation, are used to warp source image at each frame.

2.3 Detection methods for fake media

Several insights where obtained after reading deepfake detection articles. Additive random noise and video compression are the simplest improvements of fake videos indistinguishability from real ones. Decreasing the video quality, for example, by means of increasing its compression causes the accuracy of detection to decrease sometimes even if it was present in the train set. Gaussian blur and JPEG compression make fake images less distinguishable too as demonstrated in the paper [33]. There the models trained after such preprocessing had worse mean accuracy of classification compared to non using it. Paper [32] proposes a method to increase generalization of neural net models to new image generation methods. They say that adding Gaussian blur and Gaussian noise can both change low level pixel statistics, which serve well for depressing low level unstable clues. In order to increase the diversity of training samples, random extent of these preprocessings were applied. Same idea is proposed in paper [41]: the data augmentation scheme (JPEG compression, Gaussian noise, Gaussian blur, and Gamma correction) significantly improves the robustness of the detectors and meanwhile they still maintain high performance on original unaltered data. Almost the same ideas were proposed in paper [42] but with better results.

The deepfake detection methods described below are divided into two groups based on their input.

2.3.1 Single image detection

Face X-ray [22] approach is based on the feature of the blending of a face into a background when a mask delimiting the manipulated region can be created from the image. Predicting blending border is claimed to be more generalizable to new fake generation methods then classification only. The method relies on the existence of a blending step, so when an image is entirely synthetic, it is possible that it will fail. The probabilities of the input image being real or blended are calculated based on the predicted so-called X-ray image. Face Warping Artifacts [25] takes as input the rectangle crops from images that contain both the face and surrounding areas. Default architectures of CNN models, such as VGG16, ResNet50, ResNet101 and ResNet152 are trained. Resnets are fine-tuned starting from ImageNet weights, hard mining fine-tuning is employed afterwards. A video is called fake if a certain number of frames in this video are detected as fake. MesoNet [6] approach proposed two architectures. Firsts is Meso-4 network that begins with a sequence of four layers of successive convolutions and pooling, and is followed by a dense network with one hidden layer. MesoInception-4 is a similar network in which the first two convolutional layers of Meso4 are replaced by a variant of the inception module.

Visual Artifacts [26] algorithm exploits missing reflections and missing details in the eye and teeth areas. The eye region is segmented by considering the pixels in the convex hull of the associated landmarks. To segment the teeth, the image is converted to grayscale. The pixels contained in the convex hull of the mouth landmarks are clustered into a bright and dark cluster by K-Means clustering. All pixels in the bright cluster are considered as belonging to teeth. Next, the texture is used to generate features that describe the complexity of the texture. This feature vectors are than classified using small NN and logistic regression. Head Pose method [1] is based on the idea that the face border remains the same but details of face central part are changed after face swap. Head pose in the camera space helps to distinguish between the fake and real images. Firstly, a person's head mesh should be transformed from global coordinates into camera 3D space and then projected to the image plane to get the picture of his head. A standard human head mesh is used to obtain the 3D position of the head. Facial landmark points in 3D, i.e. on the head mesh and 2D, i.e. input image, are used to calculate the parameters of the necessary transformation. This is done twice, firstly using a subset of landmarks both on the face border and center and, secondly, taking only the subset of points from the face central part. After transformation parameters are calculated twice for a given image they are normalized and used as feature vectors to classify the images as real or fake using SVM with RBF kernels. Fakespotter [2] uses the fact that feature vectors that are less susceptible to adversarial attacks compared to raw pixels. A CNN backbone is used to extract features from each image. The length of calculated feature vector is equal to the number of linear and convolutional layers in the backbone and its elements are equal to the number of active neurons in the feature map of the layer. The active neuron is one with a value bigger than the threshold precomputed for this layer. The threshold for a given layer is the mean neuron value in the layer that is averaged over all images in the train set. The classifier network includes 5 FC layers and takes the above described feature vector as input.

2.3.2 Multiple frames detection

Eye Blinking [23] model takes eye crop sequences of 10-20 frames. Eye area is cropped out of each video frame, the rectangle region is generated by first extracting the bounding boxes of each eye's landmark points. The method is based on a feature of deepfake method based on GAN: while swapping faces with this generation method the eyes are copied not in the same state, for e.g. closed in the source but open in the target. So fake videos can be detected based on the frequency of blinking. Steps for this are the following: face detection, facial landmarks detection, face aligning using landmarks, feature extraction from each frame with VGG16. Next sequence learning with LSTM is done on this temporal sequence of feature maps and FC layer are used to predict if the eye is open or closed. Deepfake Video Detection Using RNN described in [24] proposed a network that is fed with a sequence of resized and normalized frames. These frames are transformed into feature maps using pretrained InceptionV3. Next, features are concatenated and passed to the LSTM. Finally, there are 2 FC layers with dropout and softmax layer. Recurrent Convolutional Strategy from [27] comprises preprocessing such as face cropping and alignment applied to a sequence of video frames. RNN is used to «exploit temporal discrepancies across frames», so the pipeline includes CNN and RNN. CNN backbone is firstly trained on the FF++ training split for binary classification to develop features to discern real faces from synthetics. This backbone is then extended with RNN, which is a GRU cell fed with feature vectors from CNN. XcepTemporal from paper [28] takes as input face crops for a sequence of frames. CNN is used to obtain a vector representation of a facial region of a frame. A sequence of such vectors is passed into a bidirectional LSTM module to learn a latent representation capable of discriminating between facial manipulations and original faces. The spatial features from the XceptionNet are passed into a first bidirectional-LSTM layer. The outputs of the first bidirectional-LSTM layer are passed to a second bidirectional-LSTM layer to produce secondary feature abstraction. The feature vector from the last LSTM unit of this second bidirectional layer is passed into a FC layer and finally to a classification layer (1 real + n fake classes). FakeCatcher's [30] input is fixed-length video clips with facial parts from input videos. Several biological signals, such as C-PPG and G-PPG, are extracted from face regions for each clip. SVM is trained on 126 features of these signals extracted from 3 regions on cheeks and nose and their transformations. Instead of a class prediction of SVM, SVR outputs the probability of a given segment to be real or synthetic. Expectations of the probabilities are used as the true threshold to classify authenticity. video class label is averaged over segments (clips) that were cut with a certain step. Capsule forensics [31] Net includes 3 primary capsules and 2 output capsules. The latent features extracted from face crops by part of the VGG-19 network are distributed to the three primary capsule blocks of CapsuleNet. The output of the algorithm is 2 vectors that are used to calculate probabilities for fake/real classification with softmax. «A video worth 1k lies» from paper [3] take input in form of a video clip. I3D is Inception model with 3D convolutions in place of 2D ones. The last layer in the network is replaced by a single neuron layer, which outputs one scalar value. NN was first pre-trained on the human action dataset and then fine-tuned on the FF++ dataset. The I3D uses pre-trained weights of the Inception model on ImageNet as its initialization.

Dynamic prototypes from paper [4] inputs a video clip's optical flow and a random frame from it. Proposed network consists of High Resolution Net (HRNet) used as encoder followed by a prototype module and a FC layer. Input is transformed into latent tensor z by HRNet. A prototype similarity vector P is calculated for z using m prototype vectors. An element in P is equal to the maximum value of inverted distance between corresponding prototype and all vectors in z. Vector P is mapped into 2 class probabilities using FC layer and softmax layer.

3 Method

3.1 Selecting baseline

Video-level and frame-level quality scores were compared on the same datasets for different models to choose the best deepfake detection model. Accuracy and ROC-AUC scores were used to choose the best model, because they are the most frequently reported scores in the deepfake detection papers. Video-level dominance was preferred to frame-level one for both scores and outperforming in AUC score was more appreciated than in accuracy. AUC is the «area under curve» which is receiver operating characteristic curve, or ROC curve. It uses the probabilities outputted from binary classifier to depict the True Positive rate against False Positive rate for different probability thresholds used to divide fakes samples from real ones. Accuracy score uses only one probability threshold value equal to 0.5 to classify samples into two classes. From the other hand, ROC-AUC allows to aggregate the TPR and FPR scores for different possible thresholds, thus provides a more robust estimate of classification performance for a model.

The best models found taking in account these rules are the following: M2TR [37], Long-dist [38], GRnet [35], XcepTemporal [28], Self. Trans. [43], LipForensics [44], I3D [3], New-CapsuleNet [39], Self-Adversarial [40], High.Freq. [34], ADT [36]. Not all deepfake detection papers published before August 2022 were included in the «Related work» section above, because the number of papers on this topic is very large and is still growing really fast. The literature review for this work was done in May 2022 and about 80 new fake detection papers appeared in Google Scholar since then. Choosing the best model is not very strait-forward thing too because various metrics are reported in the papers for different test settings.

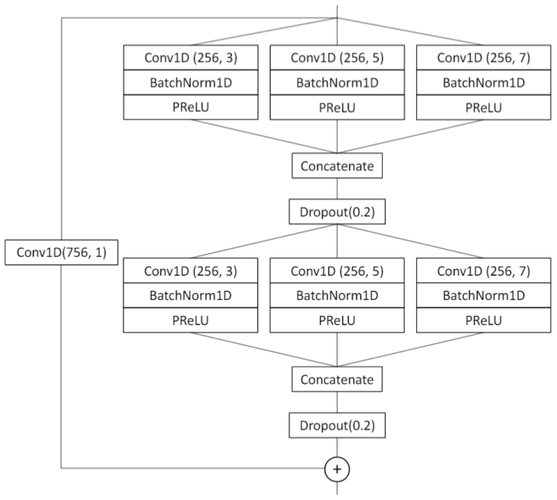
After several top-performing models were selected their publicly available implementations were used for reproduction of results reported in corresponding papers to be used as baseline. The public implementation was found for the following methods: M2TR [37], XcepTemporal [28], LipForensics [44], Self-Adversarial [40], High.

Freq. [34]. Among these models LipForensics network performs better than other ones according to the quality scores.

3.2 Description of the baseline

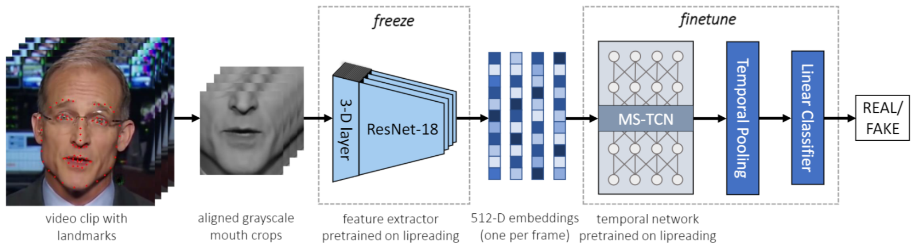
The main idea of LipForensics model, described in paper [44], is that face forgery classification can be based on distinguishing between natural and anomalous mouth movements. Authors hypothesize that irregularities in mouth motion exist in fake videos regardless of the generation method that produced them. Due to the semantically high-level nature of these cues, they are also less easily corrupted by common perturbations. The input to neural metwork is a sequence of 25 aligned gray-scale mouth crops from video frames. Pipeline starts from spatio-temporal feature extractor and is followed by temporal network. The net is first trained on the lipreading problem. After that the feature extractor is fixed and the sequence classification net is fine-tuned for binary real/fake classification with a new classification head being trained from scratch. Specifically, the feature extractor is a ResNet-18 with an initial 3-D convolutional layer which preserves the temporal dimension via padding. The feature extractor outputs a 512 dimensional vector for each input frame. The temporal net is a multi-scale temporal convolutional network (MS-TCN), combining short- and long-term temporal information at every layer by concatenating outputs of multiple branches, each with a different temporal receptive field. After a temporal global average pooling layer, a task-specific linear classifier outputs the estimated class probabilities.

Diagram of a single block in the multi-scale temporal convolutional network (MS-TCN), that was employed inside temporal part of [44], is shown below.



The stride is 1 for all convolutions. The full temporal network consists of four such blocks. The dilation rate in the first block is equal to 1, and each subsequent block's dilation rate is 2× the previous one's.

The image below depicts the schematic structure of LipForensics.



freeze
finetune
3-D layer
ResNet-18
MS-TCN
Temporal Pooling
Linear Classifier
REAL/ FAKE
video clip with landmarks
aligned grayscale mouth crops
feature extractor pretrained on lipreading
512-D embeddings (one per frame)
temporal network pretrained on lipreading

3.3 Improving the baseline

To begin with, LipForensics model was analyzed in details to understand what can be improved in this neural network. In short, the feature extractor outputs a vector per frame from input sequence, then temporal net followed by classification head classifies these multi-dimensional time-series as fake or real.

Preparing of this model includes several steps. The first one is training a multi-class neural network to perform lipreading on real videos. This training was conducted in the other research work [45] where the sota LipReading model was proposed. The LipReading net comprises a spatio-temporal feature extractor, a temporal net, and a linear classifier. This model was improved by changing the temporal part from GRU to TCN. The parameters of these subnetworks are randomly initialised and then optimised together to minimise the standard cross entropy loss. To train the forgery detector model authors replace the classifier in lip-reading model with a binary one. The temporal net is fine-tuned and the binary classifier is trained from scratch to minimize the binary cross entropy loss but feature extractor is kept fixed during this phase.

It was clear stated in the paper [44] that it is important to keep the feature extractor frozen, because the same thing was done when the LipForensics model was created from LipReading model [45]. Also it is reported in work [46] that TCN architecture outperforms GRU and LSTM in problems with sequences. Thus, the remaining alternative architecture for temporal part of Lipforensics is transformer, but that one needs comparatively big amount of data and time to train. Therefore, the main idea of improving the baseline was set to enhancing the time-series classification part of this model.

Among the papers dedicated to time-series classification improvement the approach [47] was found. Its application is very simple, authors propose to place a special transformation block before classifier to upgrade its performance.

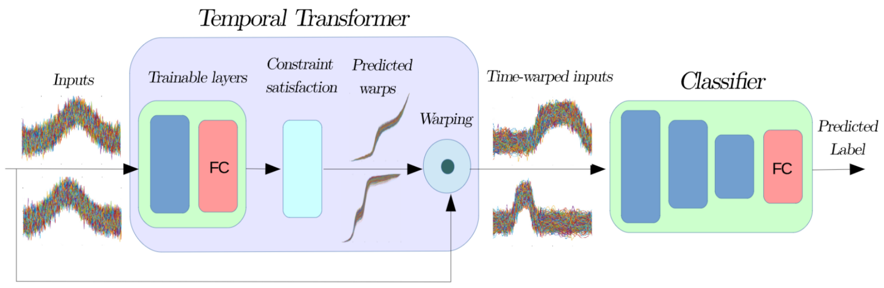
TTN stands for Temporal Transformer Networks and for an input sequence of vectors it outputs a warping function that is applied to the input to improve sequence classification. This module can be divided into three sub-modules. Trainable layers go first and their input is a sequence of T vectors with arbitrary dimension. This part of TTN includes several convolutional and fully-connected layers. The second part of TTN is the constraint satisfaction layers. Here, the output of the first submodule is normalized to get a unit-vector and then each of its entries are squared. The resulting sequence is treated as the network's estimate of the derivative of the warping function. The cumulative sum multiplied by the length of the input sequence forms the warping function. The last step in TTN module is applying differentiable temporal re-sampling where the warping function is applied to the input sequence using linear interpolation. Finally, TTN block outputs a vector of length T, such that the first element is set to zero.

In our case, this transformation was applied to the sequence of 512-dimensional vectors outputted by feature extractor for each of 25 input frames. TTN block was added inside LipForensics model right after ResNer-18 (feature extractor that outputs the sequence of per-frame vectors) and before TCN (temporal net that is fed with that sequence of vectors). The parts of the new model were initialized with weights of LipForensics, except for TTN module that was initialized with random weights.

Initial configuration for training LipForensics with TTN inside was adopted from papers [44] and [47]. The architecture for TTN block was taken from an example presented in paper [47], where the adding of TTN improved classification score on a dataset. That dataset was found to be the most non-trivial one among the described examples and includes 3D coordinates for skeleton joints. The task was action classification for skeleton movements, i.e. time-series of 3D points. Other adopted thing was the rule that weights of the TTN should be updated at one-tenth the learning rate of the classifier.

Trainable part of TTN includes 2 temporal layers with 1-dimensional convolutions with kernel sizes 8 and 16. They preserve vectors dimension equal to 512 and sequence length equal to 25, the same as the output of ResNet has that is the transformed sequence of mouth crops from 25 video frames. These two layers are followed by 3 fully-connected layers which are applied to the flattened signal matrix. The result of flattening is a 1-dimensional vector with length 512×25 that is next transformed by linear layers into vectors with sizes 16, 16 and 24, respectively. All layers in TTN are followed by Relu non-linearity. After this transformations a zero value is added to the beginning of resulting vector to get the length of input sequence. Then, this vector is normalized and numerically integrated to get 1-dimensional vector of length 25 which is used for temporal resampling of 512-dimensional time-series of feature vectors. In other words, the sequence of feature vectors is linearly interpolated and values at new time points specified in the calculated resampling vector are taken to form new transformed time-series for further classification.

The image below depicts the schematic structure of TTN.



4 Datasets

There are a lot of datasets for deepfake detection. In this work I work with FaceForensics++ [5] and CelebDF-v2 [48].

FaceForensics++ is a dataset of facial forgeries that enables researchers to train deep-learning-based approaches in a supervised fashion. The dataset contains manipulations created with four state-of-the-art methods, namely, Face2Face, FaceSwap, DeepFakes, and NeuralTextures. Four automated face manipulation methods, which are applied to 1,000 pristine videos downloaded from the Internet, specifically from YouTube. 1,000 video sequences containing 509,914 images were selected and used as pristine data. Two computer graphics-based approaches (Face2Face and FaceSwap) and two learning-based approaches (DeepFakes and NeuralTextures) were selected for the datasets. All the four methods require source and target actor video pairs as input. The final output of each method is a video composed of generated images. All the four methods were described in Related work section.

Celeb-DF is DeepFake video dataset which contains 5,639 high-quality DeepFake videos of celebrities generated using improved synthesis process. The fake video are generated with a new algorithm based on the first encoder-decoder DeepFake algorithm. Resolution of synthesized faces became bigger and color mismatch between the synthesized donor's face with the original target's face is significantly reduced. The face masks are smoother and temporal flickering of synthetic faces in the DeepFake videos is reduced by incorporating temporal correlations among the detected face landmarks.

An example of fake portraits from the CelebDF dataset is below, where the original face in on the left and the others are fake.

5 Experiments

5.1 Reproducing scores for the baseline

The only one LipForensics pretrained model was made available at the github repository [54] with the following comment: «This model has been trained on FF++ (Deepfakes, FaceSwap, Face2Face, and NeuralTextures) and is the one used to get the LipForensics video-level AUC results in Table 2 of the paper». Table 2 in paper [44] does not mention the compression level of FF++ train set. One of the authors replied in «Issues» section of their repository that the published model was trained on FaceForensics++ dataset (FF++) [5] with c23 compression level where 23 stands for the quantization number of H.264 standard of video compression.

The public model with pre-trained parameters was tested on the FF++ test set with 3 different compression levels in the same settings as described in the paper [44]. Video-level AUC obtained for c23 compression (HQ) is 98.7% that is 1% lower than the value reported in Table 4 of paper [44]. Table 4 shows the "Performance on FF++ when trained and tested on uncompressed (Raw), slightly compressed (HQ), and heavily compressed (LQ) videos". The same model achieves video-level AUC 81.5% on CelebDF-v2 dataset which is 0.9% lower than score reported in Table 2 of [44]. The decrease in about 1% of AUC for the reproduced scores can be attributed to the change of some open-source scripts used for pre-processing of the videos or some missing information that authors did not shared in their article [44].

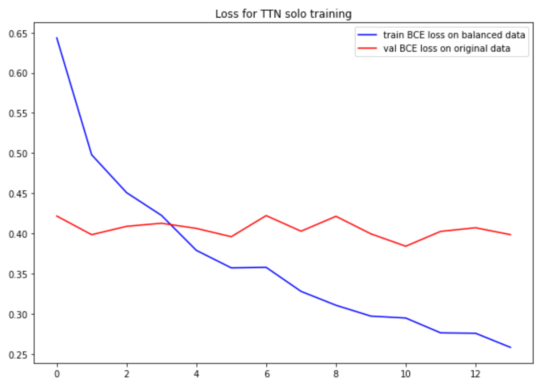
As for inference reproducing, there are no scripts provided by the authors for the preprocessing of videos except the links to third party code. Preprocessing starts from frame extraction from a video. Then face detection is applied using third party code [55] which results in bounding boxes for each frame. It was mentioned in the paper [44] that this boxes should be enlarged 1.3 times and only biggest face should be used but no code was provided for this. Next, facial landmarks detection was performed using third party code [56]. The algorithm takes a frame and a face bounding box as input and outputs a set of 68 2D points. The frame and mouth keypoints are used to crop mouth region from the image and to warp it into a standard position of keypoints. The latter action is performed using the script provided by authors of [44] as well as network inference and AUC calculation.

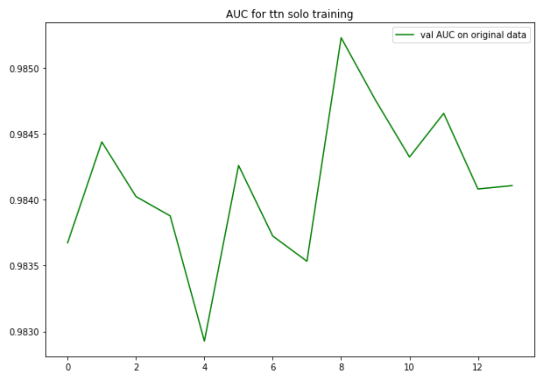
5.2 Training the proposed architecture

The training script used to train Lipforensics network [44] was not provided by the authors, so it was written from scratch. Also there was mentioned in the paper that minority class was over-sampled but the oversampling strategy itself was not mentioned, so I used oversampling with duplicates to change real to fake ratio from 1:4 to 1:1. Batch size was set to 32 as in [44], Adam optimization was used with default learning rate value $2 \times 1e-4$. As data augmentation for training, the clips were randomly cropped and horizontally flip with probability 0.5. In the same time, minority oversampling was not used for validation split.

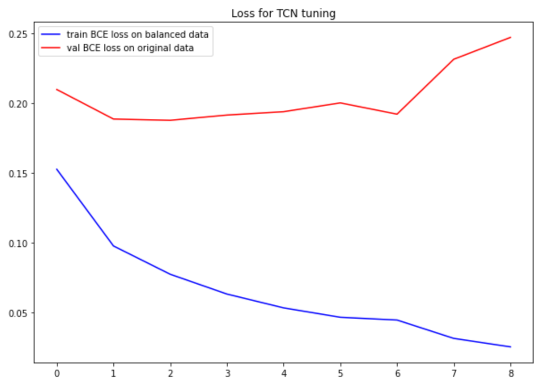
The Lipforensics model with TTN block inside was trained in two phases. Firstly, both feature extractor and temporal part of the net were kept frozen while TTN was trained. After choosing the optimal epoch which was considered to be the one with highest video-level AUC, temporal part of Lipforensics was unfrozen. This time training starts from the optimized ckeckpoint and the feature extractor is fixed again. Such a strategy was preferred in order to not change the parameters of TCN network too much because it was fine-tuned from LipReading model's weights of TCN during the training of LipForensics model. This idea was also highlighted in [44] as well as necessity to keep the feature extractor fixed. So the proposed training strategy was chosen to not change the parameters of the sequence classifier too much. One epoch of training takes 1h 30min on Tesla V100 during training LipForensics with TTN on balanced train split of FF++ dataset. The training plots for different hyper-parameters are presented below with comments of what is changed in comparison with default settings described above. Each training experiment has a number to easier refer to them.

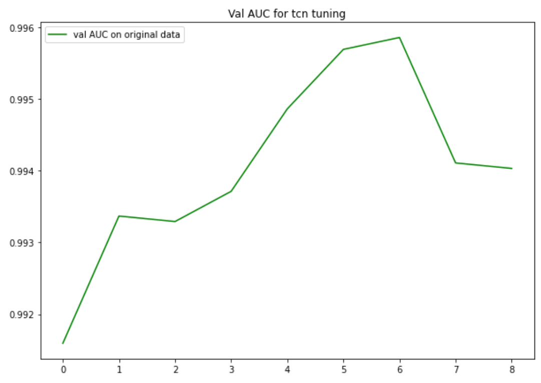
c0.1) Solo TTN training on train set balanced by minority oversampling with duplicates (real:fake = 1:1) and c0, i.e. raw, compression. Learning rate (lr) for TTN is equal to 2*1e-4/10. The calculated scores per epoch are presented in the images below.



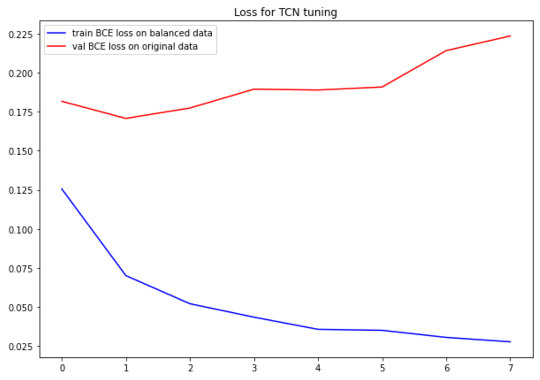


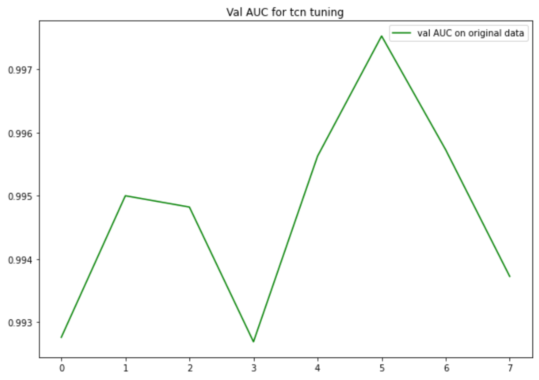
c0.1.a) Tuning of TTN with TCN starting epoch 8 in the previous case that has the highest validation AUC. Lr for TCN set to 2*1e-5.



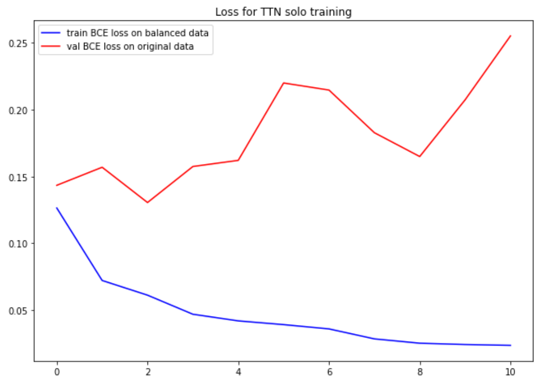


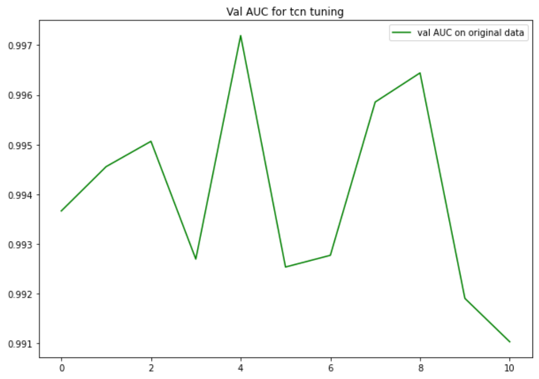
c0.1.b) Here the same settings as in previous example but lr for TCN is taken equal to 2*1e-4/3.





c0.1.c) Again only the lr for TCN is changed to 2*1e-4.

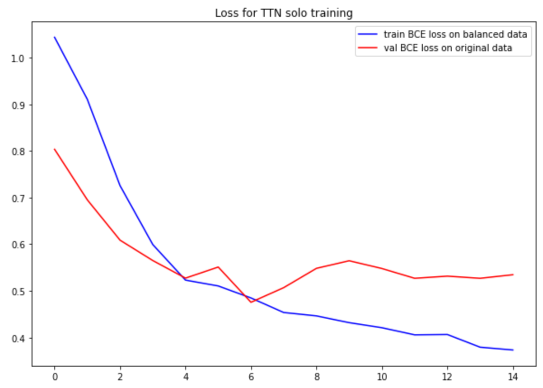


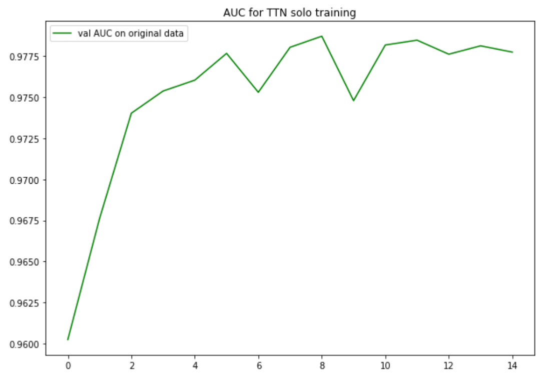


As it can be noticed in the three last examples, training TCN with TTN tends to overfit after 2-3 epochs of training when TCN is not fixed anymore. Nonetheless, validation AUC does not follow the same trend as validation loss and gets bigger at some epochs even if these have bigger val loss.

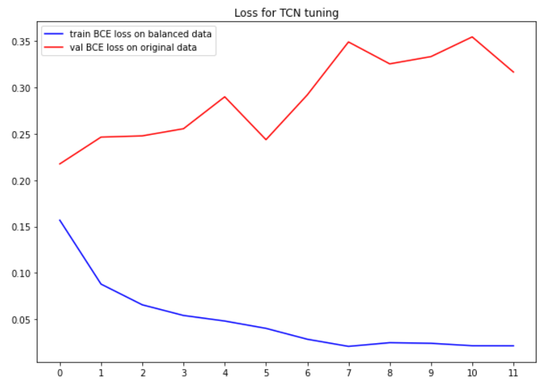
FF++ dataset used before for train and validation was raw uncompressed data. Although the AUC score was improved for FF++ with all three compression levels, this did not happen for CelebDF-v2 as can be seen in Table 1. Actually, raw FF++ was used because it was not found out at that moment that authors used c23 data to train their model and calculate performance scores for LipForensics net that were reported in Table 2 of the paper [44]. So we switched to c23 data to have the same experiment settings as authors did for a proper evaluation of the impact of proposed changes in the network architecture. c23 video compression also seems to be closer to data in CelebDF-v2 that was collected from Youtube.

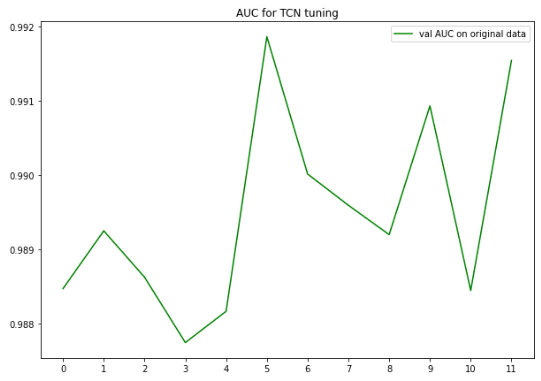
c23.1) Solo TTN training was performed on balanced train set with c23 compression. TTN lr was set to 2*1e-4/10.



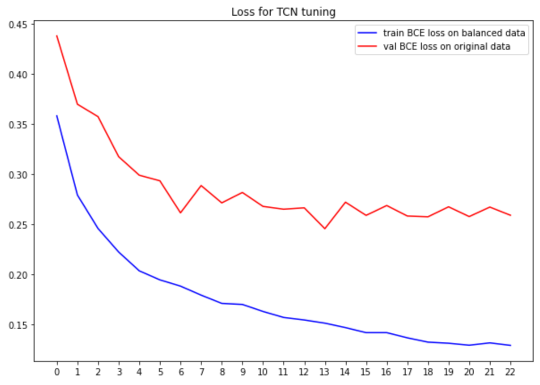


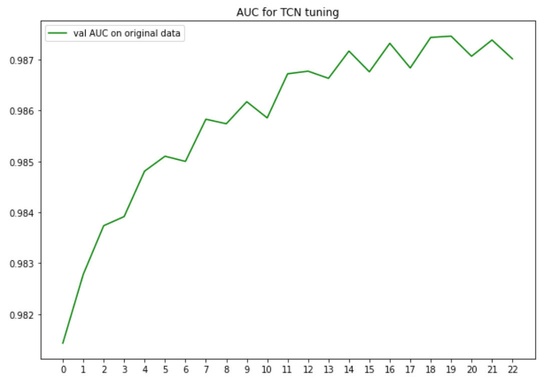
c23.1.a) Next training of TTN with TCN starting from params for epoch 8 is done. Again with the highest validation AUC. Lr for TCN = 2*1e-4.





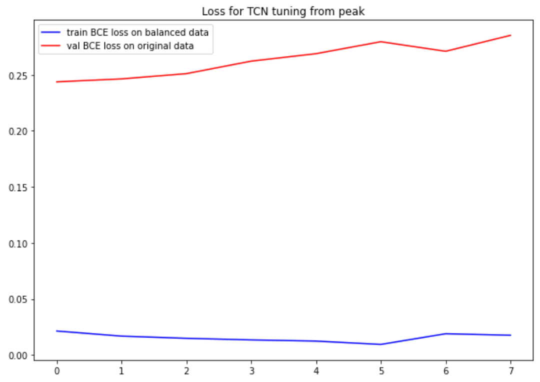
c23.1.b) The loss plots for training both TTN and TCN gives an idea that learning rate is excessively big, but actually maximum values in AUC plots are bigger for bigger lr. This time loss was decreased from 1e-6 to 1e-7 during training when val loss decrease stagnated. Training started again from epoch 8 with the highest validation AUC and TTN lr kept 10 times smaller.

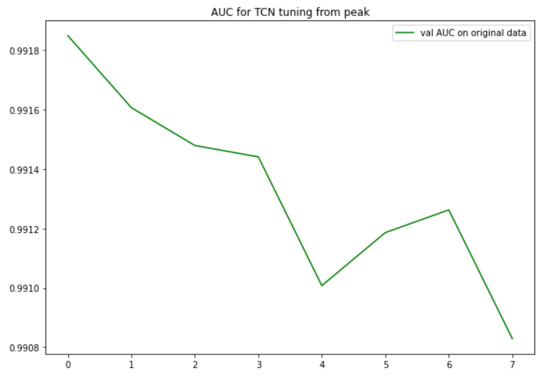




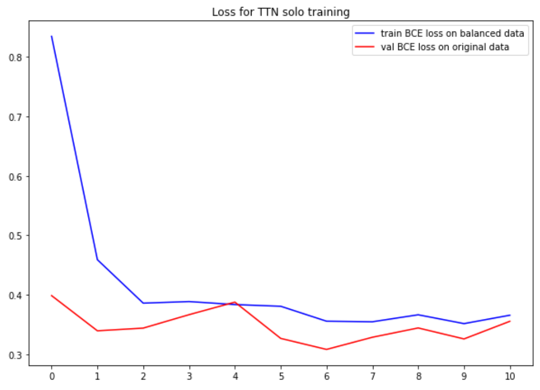
Decreasing LR from 2*1e-4 to 1e-6 for simultaneous training of TTN and TCN solved overfitting that was present in all same previous training experiments (other settings remained unchanged). The obtained AUC stops growing and models trained on raw FF++ data were not outperformed by models trained on c23 data in this experiment as can be seen in Table 1.

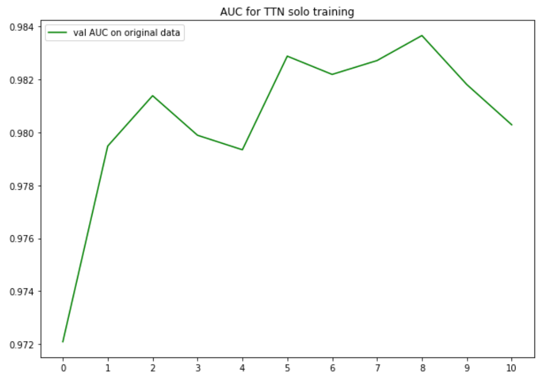
c23.1.c) So the convergenges for smaller learning rate seemed to be much more stable. I wanted to exploit this and started training TTN with TCN from epoch 5 in experiment 6 with the highest validation AUC, but this time with smaller lr: TCN lr was 1e-6.



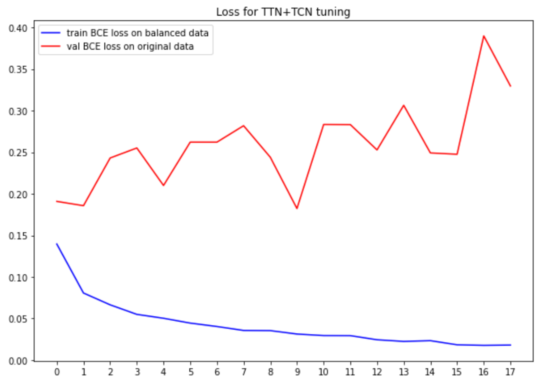


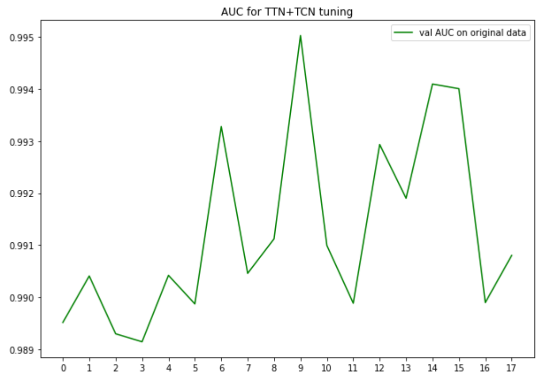
c23.2) The model trained on raw data still was not outperformed so I tried a new initialization for the trainable layers in the TTN module. It was initialized to perform identity transformation that is quiet logical because when the other parts of LipForensics are kept fixed the temporal classifier cannot perform with same accuracy on new feature vectors.





c23.2.a) Strating from ep 8 in previous experiment TCN and TTN were trained with TCN lr equal to 2*1e-4.





5.3 Results

Video-level AUC scores obtained in the most successful experiments described before are presented in the Table 1 below. Rows correspond to model checkpoints and columns correspond to test splits of the FF++ [5] dataset with different compression levels (c0, c23, c40) and [48] dataset. Models trained under this project have the specified code of experiment. The compression level of train set is also indicated for each model. The scores reported in the paper [44] in Tables 2 and 4 are also included for comparison and are called «LipForensics (reported)». Baseline scores are considered to be the ones reproduced by us though, because all the possible information was taken into account but it still did not lead to exactly the same reproduced scores. The reproduced scores are listed in italic in the row with model named «LipForensics (public ckpt), c23».

Model \ Test splits	FF++, c0	FF++, c23	FF++, c40	CelebDF-v2
LipForensics (reported), c0	99.9	-	-	-
LipForensics (reported), c23	-	99.7	-	82.4
LipForensics (public ckpt), c23	*98.9*	*98.7*	*81.3*	*81.5*
LipForensics+TTN, c0.1.c)	99.7	99.5	83.7	78.9
LipForensics+TTN, c23.2.a)	**99.6**	**99.6**	**84.0**	**81.4**

Table 1: Comparison of video-level AUC scores.

As we can see adding TTN module leads to improving the models performance on FF++ but that is not the case for CelebDF-v2 for which only approximately the same score was obtained with TTN. This effect is expected because making neural net deeper and fine-tuning it on the same dataset, which it was trained on before, pushes it to overfit on train dataset, i.e. lose generalization to new datasets. The final result of this work is the model «LipForensics+TTN, c23» in the last row of the table that outperforms the baseline model «LipForensics (public ckpt), c23». For reproducibility, the seeds of all random modules was set to 42, cudnn.deterministic was set to True and cudnn.benchmark to False in torch module both during training and inference of the models. The code used for training and inference can be found at my github [57].

6 Conclusion

Fake media detection for liveness verification pipeline was studied in this research work. This domain is very fast growing and relevant for modern cybersecurity domain. An improved LipForensics model was obtained by adding a TTN module inside it to boost time-series classification part of the model while keeping the feature extraction part of the model intact. Even though the scores obtained for model with proposed architecture change is sometimes smaller than the reported ones in [44], the adding of TTN module is improving the scores independently of compression level of FF++ data. The challenge appears with other dataset, not presented in the train set or validation set, as always happens in ML. So, this problem can be a good direction for future work, because it is important to keep a deepfake detection algorithm ready to catch new photos and videos created with unknown manipulation techniques. The solution for network generalization to CelebDF dataset after training on FF++ dataset was presented in papers [41] and [42]. Authors improve the increase of quality on the Celeb-DF test set due to distortion augmentation of the FF++ train set while preserving the same quality on FF++ test.

Other ideas that can be tested in future research is using 2 parallel TTNs and concatenating the TTN outputs results. Of course, using transformer architecture trained from scratch is also a possible continuation of this research, but this approach would probably require training the 3DConv-ResNet-Transfomer-FC architecture on lip-reading task first and then fine-tuning the temporal part of the net with a new 2-class FC head on the binary classification task.

7 Bibliography

[1] Yang, X., Li, Y. and Lyu, S., 2019, May. Exposing deep fakes using inconsistent head poses. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 8261-8265). IEEE.

[2] Wang, R., Juefei-Xu, F., Ma, L., Xie, X., Huang, Y., Wang, J. and Liu, Y., 2019. Fakespotter: A simple yet robust baseline for spotting ai-synthesized fake faces. arXiv preprint arXiv:1909.06122.

[3] Wang, Y. and Dantcheva, A., 2020, November. A video is worth more than 1000 lies. Comparing 3DCNN approaches for detecting deepfakes. In 2020 15Th IEEE international conference on automatic face and gesture recognition (FG 2020) (pp. 515-519). IEEE.

[4] Trinh, L., Tsang, M., Rambhatla, S. and Liu, Y., 2021. Interpretable and trustworthy deepfake detection via dynamic prototypes. In Proceedings of the IEEE/CVF winter conference on applications of computer vision (pp. 1973-1983).

[5] FaceForensics++ Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J. and Nießner, M., 2019. Faceforensics++: Learning to detect manipulated facial images. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 1-11).

[6] Afchar, D., Nozick, V., Yamagishi, J. and Echizen, I., 2018, December. Mesonet: a compact facial video forgery detection network. In 2018 IEEE international workshop on information forensics and security (WIFS) (pp. 1-7). IEEE.

[7] Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C. and Nießner, M., 2016. Face2face: Real-time face capture and reenactment of rgb videos. In Proceedings

of the IEEE conference on computer vision and pattern recognition (pp. 2387-2395).

[8] Thies, J., Zollhöfer, M. and Nießner, M., 2019. Deferred neural rendering: Image synthesis using neural textures. ACM Transactions on Graphics (TOG), 38(4), pp.1-12.

[9] Kim, H., Garrido, P., Tewari, A., Xu, W., Thies, J., Niessner, M., Pérez, P., Richardt, C., Zollhöfer, M. and Theobalt, C., 2018. Deep video portraits. ACM Transactions on Graphics (TOG), 37(4), pp.1-14.

[10] Nguyen, T.T., Nguyen, Q.V.H., Nguyen, D.T., Nguyen, D.T., Huynh-The, T., Nahavandi, S., Nguyen, T.T., Pham, Q.V. and Nguyen, C.M., 2022. Deep learning for deepfakes creation and detection: A survey. Computer Vision and Image Understanding, p.103525.

[11] Perov, I., Gao, D., Chervoniy, N., Liu, K., Marangonda, S., Umé, C., Dpfks, M., Facenheim, C.S., RP, L., Jiang, J. and Zhang, S., 2020. DeepFaceLab: Integrated, flexible and extensible face-swapping framework. arXiv preprint arXiv:2005.05535.

[12] Ha, S., Kersner, M., Kim, B., Seo, S. and Kim, D., 2020, April. Marionette: Few-shot face reenactment preserving identity of unseen targets. In Proceedings of the AAAI conference on artificial intelligence (Vol. 34, No. 07, pp. 10893-10900).

[13] Deng, Y., Yang, J., Chen, D., Wen, F. and Tong, X., 2020. Disentangled and controllable face image generation via 3d imitative-contrastive learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 5154-5163).

[14] Tewari, A., Elgharib, M., Bharaj, G., Bernard, F., Seidel, H.P., Pérez, P., Zollhofer, M. and Theobalt, C., 2020. Stylerig: Rigging stylegan for 3d control over portrait images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6142-6151).

[15] Li, L., Bao, J., Yang, H., Chen, D. and Wen, F., 2019. Faceshifter: Towards high fidelity and occlusion aware face swapping. arXiv preprint arXiv:1912.13457.

[16] Siarohin, A., Lathuilière, S., Tulyakov, S., Ricci, E. and Sebe, N., 2019. First order motion model for image animation. Advances in Neural Information Processing Systems, 32.

[17] Wiles, O., Koepke, A. and Zisserman, A., 2018. X2face: A network for controlling face generation using images, audio, and pose codes. In Proceedings of the European conference on computer vision (ECCV) (pp. 670-686).

[18] Zakharov, E., Shysheya, A., Burkov, E. and Lempitsky, V., 2019. Few-shot adversarial learning of realistic neural talking head models. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 9459-9468).

[19] Averbuch-Elor, H., Cohen-Or, D., Kopf, J. and Cohen, M.F., 2017. Bringing portraits to life. ACM Transactions on Graphics (ToG), 36(6), pp.1-13.

[20] Hong, F.T., Zhang, L., Shen, L. and Xu, D., 2022. Depth-Aware Generative Adversarial Network for Talking Head Video Generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3397-3406).

[21] Nirkin, Y., Masi, I., Tuan, A.T., Hassner, T. and Medioni, G., 2018, May. On face segmentation, face swapping, and face perception. In 2018 13th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2018) (pp. 98-105). IEEE.

[22] Li, L., Bao, J., Zhang, T., Yang, H., Chen, D., Wen, F. and Guo, B., 2020. Face x-ray for more general face forgery detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 5001-5010).

[23] Li, Y., Chang, M.C. and Lyu, S., 2018, December. In ictu oculi: Exposing ai created fake videos by detecting eye blinking. In 2018 IEEE International workshop on information forensics and security (WIFS) (pp. 1-7). IEEE.

[24] Güera, D. and Delp, E.J., 2018, November. Deepfake video detection using recurrent neural networks. In 2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS) (pp. 1-6). IEEE.

[25] Li, Y. and Lyu, S., 2018. Exposing deepfake videos by detecting face warping artifacts. arXiv preprint arXiv:1811.00656.

[26] Matern, F., Riess, C. and Stamminger, M., 2019, January. Exploiting visual artifacts to expose deepfakes and face manipulations. In 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW) (pp. 83-92). IEEE.

[27] Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I. and Natarajan, P., 2019. Recurrent convolutional strategies for face manipulation detection in videos. Interfaces (GUI), 3(1), pp.80-87.

[28] Chintha, A., Thai, B., Sohrawardi, S.J., Bhatt, K., Hickerson, A., Wright, M. and Ptucha, R., 2020. Recurrent convolutional structures for audio spoof and video deepfake detection. IEEE Journal of Selected Topics in Signal Processing, 14(5), pp.1024-1037.

[29] Agarwal, S., Farid, H., El-Gaaly, T. and Lim, S.N., 2020, December. Detecting deep-fake videos from appearance and behavior. In 2020 IEEE international workshop on information forensics and security (WIFS) (pp. 1-6). IEEE.

[30] Ciftci, U.A., Demir, I. and Yin, L., 2020. Fakecatcher: Detection of synthetic portrait videos using biological signals. IEEE transactions on pattern analysis and machine intelligence.

[31] Nguyen, H.H., Yamagishi, J. and Echizen, I., 2019, May. Capsule-forensics: Using capsule networks to detect forged images and videos. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 2307-2311). IEEE.

[32] Xuan, X., Peng, B., Wang, W. and Dong, J., 2019, October. On the generalization of GAN image forensics. In Chinese conference on biometric recognition (pp. 134-141). Springer, Cham.

[33] Wang, S.Y., Wang, O., Zhang, R., Owens, A. and Efros, A.A., 2020. CNN-generated images are surprisingly easy to spot... for now. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 8695-8704).

[34] Luo, Y., Zhang, Y., Yan, J. and Liu, W., 2021. Generalizing face forgery detection with high-frequency features. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 16317-16326).

[35] Guo, Z., Yang, G., Chen, J. and Sun, X., 2022. Exposing Deepfake Face Forgeries with Guided Residuals. arXiv preprint arXiv:2205.00753.

[36] Wang, P., Liu, K., Zhou, W., Zhou, H., Liu, H., Zhang, W. and Yu, N., 2022, May. ADT: Anti-Deepfake Transformer. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 2899-1903). IEEE.

[37] Wang, J., Wu, Z., Ouyang, W., Han, X., Chen, J., Jiang, Y.G. and Li, S.N., 2022, June. M2tr: Multi-modal multi-scale transformers for deepfake detection. In Proceedings of the 2022 International Conference on Multimedia Retrieval (pp. 615-623).

[38] Lu, W., Liu, L., Luo, J., Zhao, X., Zhou, Y. and Huang, J., 2021. Detection of deepfake videos using long distance attention. arXiv preprint arXiv:2106.12832.

[39] Stanciu, D.C. and Ionescu, B., 2022, June. Uncovering the Strength of Capsule Networks in Deepfake Detection. In Proceedings of the 1st International Workshop on Multimedia AI against Disinformation (pp. 69-77).

[40] Chen, L., Zhang, Y., Song, Y., Liu, L. and Wang, J., 2022. Self-supervised Learning of Adversarial Example: Towards Good Generalizations for Deepfake Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 18710-18719).

[41] Lu, Y. and Ebrahimi, T., 2022. A New Approach to Improve Learning-based Deepfake Detection in Realistic Conditions. arXiv preprint arXiv:2203.11807.

[42] Lu, Y., Luo, R. and Ebrahimi, T., 2022. A Novel Framework for Assessment of Learning-based Detectors in Realistic Conditions with Application to Deepfake Detection. arXiv preprint arXiv:2203.11797.

[43] Zhao, H., Zhou, W., Chen, D., Zhang, W. and Yu, N., 2022. Self-supervised Transformer for Deepfake Detection. arXiv preprint arXiv:2203.01265.

[44] Haliassos, A., Vougioukas, K., Petridis, S. and Pantic, M., 2021. Lips don't lie: A generalisable and robust approach to face forgery detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 5039-5049).

[45] Martinez, B., Ma, P., Petridis, S. and Pantic, M., 2020, May. Lipreading using temporal convolutional networks. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 6319-6323). IEEE.

[46] Bai, S., Kolter, J.Z. and Koltun, V., 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271.

[47] Lohit, S., Wang, Q. and Turaga, P., 2019. Temporal transformer networks: Joint learning of invariant and discriminative time warping. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 12426-12435).

[48] Li, Y., Yang, X., Sun, P., Qi, H. and Lyu, S., 2020. Celeb-df: A large-scale challenging dataset for deepfake forensics. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 3207-3216).

[49] https://liveness.com/

[50] https://github.com/deepfakes/

[51] https://github.com/shaoanlu/fewshot-face-translation-GAN

[52] https://github.com/dfaker/df

[53] https://github.com/MarekKowalski/FaceSwap/

[54] https://github.com/ahaliassos/LipForensics/

[55] https://github.com/biubug6/Pytorch_Retinaface

[56] https://github.com/1adrianb/face-alignment

[57] https://github.com/k0l1ka/deepfake_video_detection