Interpretable and Trustworthy Deepfake Detection via Dynamic Prototypes

Loc Trinh, Michael Tsang, Sirisha Rambhatla, Yan Liu
University of Southern California
Los Angeles, CA 90089
{loctrinh, tsangm, sirishar, yanliu.cs}@usc.edu

Abstract

In this paper we propose a novel human-centered approach for detecting forgery in face images, using dynamic prototypes as a form of visual explanations. Currently, most state-of-the-art deepfake detections are based on black-box models that process videos frame-by-frame for inference, and few closely examine their temporal inconsistencies. However, the existence of such temporal artifacts within deepfake videos is key in detecting and explaining deepfakes to a supervising human. To this end, we propose Dynamic Prototype Network (`DPNet`*) – an interpretable and effective solution that utilizes dynamic representations (i.e., prototypes) to explain deepfake temporal artifacts. Extensive experimental results show that* `DPNet` *achieves competitive predictive performance, even on unseen testing datasets such as Google's DeepFakeDetection, DeeperForensics, and Celeb-DF, while providing easy referential explanations of deepfake dynamics. On top of* `DPNet`*'s prototypical framework, we further formulate temporal logic specifications based on these dynamics to check our model's compliance to desired temporal behaviors, hence providing trustworthiness for such critical detection systems.*

1. Introduction

While artificial intelligence (AI) plays a major role in revolutionizing many industries, it has also been used to generate and spread malicious misinformation. In this context, *Deepfake* videos – which can be utilized to alter the identity of a person in a video – have emerged as perhaps the most sinister form of misinformation, posing a significant threat to communities around the world [63, 26, 60, 62], especially with election interference or nonconsensual fake pornography [9, 24]. Therefore, as deepfakes become more pervasive, it is critical that there exists algorithms that can ascertaining the trustworthiness of online videos.

To address this challenge, a series of excellent works has been conducted on detecting deepfakes [61, 41, 1, 49, 33]. While these work have achieved good progress towards the prediction task to a certain extent, there is still significant room for improvement. First, even though existing work focus on the detection problem, very few of them address the interpretability and trustworthiness aspects. Currently, most existing solutions draw bounding boxes around a face and label it with *fakeness* probabilities. Rather, it might be more fruitful to explain *why* a model predicts a certain face as real or fake, such as which parts of the face the model believes are forged, and where is it looking to yield this prediction. This is crucial for a human to understand and trust the content verification systems. Second, it is known that humans can instantaneously detect deepfake videos after observing certain unnatural dynamics, due to the distortions induced by deepfake generative models, which are generally harder to hide [31, 47, 75]. This too would also be a viable explanation for a system to return as humans can quickly see and understand abnormal movements (Figure 1). Yet most state-of-the-art deepfake detection techniques only analyze a potential video frame-by-frame, and few have explored these temporal inconsistencies [50, 48, 40]. As a result, there is a need for an interpretable deepfake detection method that *both* considers temporal dynamics and at the same time provides human-accessible explanations and insights into the inconsistencies within deepfake videos.

To this end, we propose `DPNet` – an interpretable prototype-based neural network that captures dynamic features, such as unnatural movements and temporal artifacts, and leverages them to explain *why* a particular prediction was made. Specifically, `DPNet` works by first learning the prototypical representations of the temporal inconsistencies within the latent space, by grouping the patch-wise representation of real video closer together while pushing those of fake videos farther away. Then, it makes predictions based on the similarities between the dynamics of a test video and a small set of learned dynamic prototypes. Lastly, the prototypes are then intermittently projected to the closest representative video patch from the training dataset, which yields an immediate human-understandable interpretation of the learned dynamic prototypes.

The primary advantages of `DPNet` are as follows:

- **Faithful explanations via case-based reasoning**: `DPNet` follows a case-based reasoning approach that

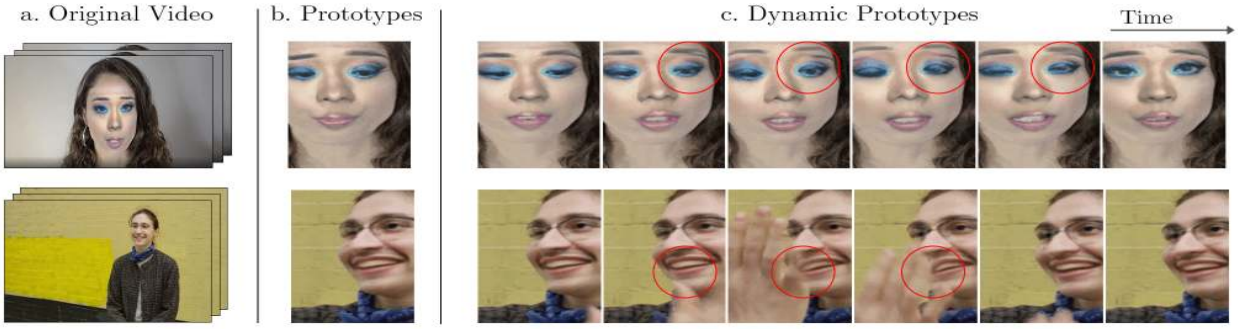


Figure 1. **Examples of static vs. dynamic explanations for deepfake videos.** Qualitatively, seeing temporal artifacts allow a human to quickly determine whether a video is real or fake. Red circles indicate regions of interest. Best view as GIFs (See Appendix C).

utilizes previously learned dynamics - as a piece of evidence (i.e *cases*) - to tackle an unseen testing video. This also allows the model to explain *why* a certain prediction was made, in a way that is reflective of the network's underlying computational process.

- **Visual dynamic explanations**: `DPNet` provides explanations in the form of visual dynamics (video clips) via the learned dynamic prototypes, each of which points to a temporal artifact that is accessible and easy for humans to understand.
- **Temporal logic specifications**: Lastly, the dynamic prototypes learned by the network can additionally be used to formulate temporal logic specifications. This allows auditors to check the robustness of the model and verify whether certain temporal behaviors are obeyed throughout the lengths of the videos.

2. Related Work

2.1. Face forgery detection

Early face forensic work focus on hand-crafting facial features, such as eye color and missing reflections [41], 3D head poses [70], and facial movements [2, 6]. However, these approaches do not scale well to larger and more sophisticated deepfakes. To address this problem, researchers leverage recent advances in deep learning to automatically extract discriminative features for forgery detection [49, 45, 44, 67]. Previous work achieved state-of-the-arts by fine-tuning ImageNet-based model, such as Xception [49]. Other work examine spatial pyramid pooling module to detect resolution-inconsistent facial artifacts DSP-FWA [35], low-level features and convolutional artifacts [1, 11], or blending artifacts via Face X-ray [33]. FakeSpotter [67] uses layer-wise neuron behaviors as features instead of final-layer neuron output to train a classifier.

Most forgery detection methods process deepfake videos frame-by-frame, and few explore multi-modal and temporal dynamics [74, 4]. Recent work using multi-frame inputs [50, 40] and video architecture [48] have shown the competitive potential of leveraging temporal information. Our approach builds on this and examines temporal artifacts both to predict and explain deepfakes to human decision makers.

With more advanced deepfake creations, recent works [13, 29, 15] have shown that the performance of current methods *drops* drastically on new types of facial manipulations. In particular, ForensicTransfer [13] proposes an autoencoder-based neural network to transfer knowledge between different but related manipulations. Face X-ray [33] created a blending dataset to help networks generalize across various manipulations, and [40] creates a novel loss to reshape the latent space, pulling real representations closer and repelling fake representations farther away, both of which have demonstrated valuable generalizability.

2.2. Interpretable neural networks

One prominent approach to explaining deep neural networks is *posthoc* analysis via gradient [54, 52, 56, 5, 53] and perturbation-based methods [19, 51, 71, 43]; however, it is known that these methods do not modify the complex underlying architecture of the network. Instead, another line of research tries to build networks that are interpretable by design, with a *built-in* way to self-explain [3, 34]. The advantage of this approach is that interpretability is represented as *units* of explanation - general concepts and not necessarily raw inputs. This can be seen in the work of Alvarez et. al [3] for basis concept learning and Kim et. al [30, 42] for case-based reasoning and prototype learning. Recently, Chen et al. [12] proposed learning prototypes for fine-grained image classification to make predictions based on similarity to class-specific image patches via ProtoPNet.

On the other hand, although complex deep learning-based video classification models have been developed for video understanding, such as 3D CNN, TSN, TRN, and more [73, 38, 73, 66, 18], there is much to be desired in terms of interpretability, especially when compared to intrinsically interpretable models. In contrast, our proposed approach extends fine-grain classification [12] and captures fake temporal artifacts as dynamic prototypes, which can be directly visualize to explain predictions to a human being, which is crucial important for face forgery detection.

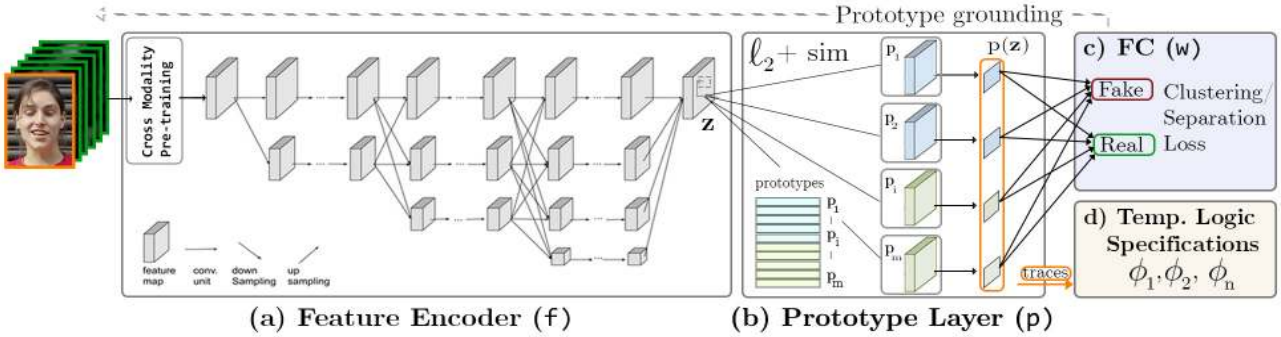


Figure 2. **DPNet video-based face manipulation detection architecture.** Spatial and temporal information are processed via the HRNet feature encoding backbone. The networks learn m prototypes, each is used to represent some prototypical activation pattern in a patch of the convolutional feature maps, which in turn corresponds to some prototypical dynamic patch in the original spatial/temporal space. Traces from videos along with prototypical activations are then verified via the temporal logic module post-training.

2.3. Safety verification

The importance of safety verification, especially in critical domains such as healthcare or autonomous driving, was highlighted when the discovery of adversarial attacks [57, 22] prompted many lines of work in robust verification in ML [32, 28, 23, 25, 68, 10, 39]. To further reason about safety and robustness in time, temporal logic has been broadly in previous work [8, 46, 55, 17], and recent work in using temporal logic to verify time-series and NN-based perception system have shown promises [64, 14, 69]. Moreover, the difficulty in building neuro-symbolic system is the question of how to integrate the output representations of deep networks in a propositional form [7]. In contrast, the dynamic prototypes from our interpretable models provide a convenient vehicle for us to specify atomic propositions and formulate temporal logic specifications for videos. This allows users verify the model's compliance to desired temporal behaviors and establish trust within critical detection systems when deployed in a real-world scenario.

3. Dynamic Prototype Network (DPNet)

In this section, we introduce our Dynamical Prototype Network (DPNet), the loss function, and the training procedure, and the logic syntax. In addition, we highlight the steps that our network took to predict a new video, and how those steps can be interpreted in a human-friendly way.

3.1. DPNet architecture

The proposed architecture is shown in Figure 2. As shown, DPNet consists of four components: the feature encoder f, the prototype layer p, the fully-connected layer w, and the temporal logic verifier. Formally, let $\mathcal{V} = \{(\mathbf{v}_i, y_i)\}_{i=1}^N$ be the video dataset, where $\mathbf{v}_i$ be a deepfake video sequence of length $T_{\mathbf{v}_i}$, and $y_i \in \{0, 1\}$ is the label for fake/real. Clips are sampled as inputs to the network.

Feature encoder $f(\cdot)$: The feature encoder f encodes a processed video input $\mathbf{x}_i \in \mathbb{R}^{256\times256\times S}$ into a hidden representation $\mathbf{z} \in \mathbb{R}^{H\times W\times C}$. We used HRNet here as the backbone encoder, initialized with ImageNet pretrained weights. The input $\mathbf{x}_i$ to the encoder f is formed by stacking one RGB frame with precomputed optical flow fields between several consecutive frames, yielding S channels (Figure 2a). We let the input to the DPNet be a fixed-length $T < T_{\mathbf{v}_i}$, and randomly selected the initial starting frame for $\mathbf{x}_i$. This allows us to explicitly describe the motion of facial features between video frames, while *simultaneously* presenting the RGB pixel information to the network. Furthermore, since the optical flow fields can also be viewed as image channels, they are well suited for image-based convolutional networks. The feature encoder f outputs a convolutional tensor $\mathbf{z} = f(\mathbf{x}_i)$ that is forwarded to the prototype layer.

Prototype layer $p(\cdot)$: The network learns m prototype vectors $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m$ of shape $(1, 1, C)$ in the latent space, each will be used to represent some dynamical prototypical activation pattern within the convolutional feature maps. The prototype layer p computes the squared ℓ_2 distance between each the prototype vectors $\mathbf{p}_j$ and each spatial/temporal *patch* (of shape $(1, 1, C)$) within the input feature maps $\mathbf{z}$. It then inverts the distance to score their similarity. This generates m similarity maps for each dynamic prototype, and max-pooling yields the most similar activation patch. The shape of the prototype vectors is chosen to represent the smallest facial dynamic patch within $\mathbf{z}$. Further patch projection in section 3.2.2 attributes each prototype to a corresponding prototypical video patch in the original pixel/flow space. Formally, in Figure 2b, the prototype layer p computes m similarity scores:

$$p(\mathbf{z}) = [\, p_1(\mathbf{z}),\ p_2(\mathbf{z}),\ \ldots,\ p_m(\mathbf{z})\,]^\top \tag{1}$$

where, the similarity score between a prototype $\mathbf{p}_j$ and $\mathbf{z}$, denoted as $p_j(\cdot)$ is given by

$$p_j(\mathbf{z}) = \max_{\mathbf{z}' \in \text{patches}(\mathbf{z})} \frac{1}{1 + \|\mathbf{z}' - \mathbf{p}_j\|_2^2} \tag{2}$$

Fully-connected layer $w(\cdot)$: This layer computes weighted sums of similarity scores, $\mathbf{a} = \mathbf{W}\, p(\mathbf{z})$, where $\mathbf{W} \in \mathbb{R}^{K \times m}$ are the weights, and K denotes the number of classes ($K = 2$ for DPNet). We then use a softmax layer to compute the predicted probability as follows, $\hat{y}_i = \frac{\exp(a_i)}{\sum_{j=1}^{K} \exp(a_j)}$. Note that, we allocate m_k prototypes for each class $k \in \{0, 1\}$ s.t. $\sum_k m_k = m$. In other words, every class is represented by m_k prototypes in the final model.

3.2. Learning objective

We aim to learn meaningful forgery representation which ensures that the dynamic prototype vectors a) are close to the input video patches (**Fidelity**), b) between fake and real artifacts are well-separated (**Separability**), and c) are interpretable by humans (**Grounded**). We adopt two widely used loss functions in prototype learning to enforce fidelity and separability. Furthermore, we also introduce a *diversity* loss term to ensure that intra-class prototypes are non-overlapping. We jointly optimize the feature encoder f along with the prototype vectors $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m$ in the prototype layer p to minimize the the cross-entropy loss on training set, while also regularizing for the desiderata.

3.2.1 Loss function

Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ be our training dataset, where $\mathbf{x}_i$ is our stacked input extracted from video $\mathbf{v}_i$.

For hyperparameters λ_c, λ_s, and λ_d the overall objective function that we wish to minimize is given by:

$$\begin{aligned} \mathcal{L}(\mathcal{D}; \theta) = CE(\mathcal{D}; \theta) + \lambda_c R_{clus}(\mathcal{D}; \theta) \\ + \lambda_s R_{sep}(\mathcal{D}; \theta) + \lambda_d R_{div}(\theta) \end{aligned} \tag{3}$$

where $CE(\cdot)$, $R_{clus}(\cdot)$, $R_{sep}(\cdot)$, $R_{div}(\cdot)$ are the cross-entropy, clustering, separation, and diversity loss, respectively. Here, θ are the trainable parameters for the feature encoder f and the prototype layer p. The cross-entropy loss here imposes prediction accuracy and is given by

$$CE(\mathcal{D}; \theta) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} -\mathbb{1}\left[y_i = k\right] \log(\hat{y}_k) \tag{4}$$

The clustering loss R_{clus} minimizes the squared ℓ_2 distance between some latent patch within a training video and its closest prototype vector from that class, and is given by

$$R_{clus}(\cdot) = \frac{1}{N} \sum_{i=1}^{N} \min_{\mathbf{p}_j \in \mathbf{P}_{y_i}} \min_{\mathbf{z} \in \text{patches}(\mathbf{x}_i)} \|\mathbf{z} - \mathbf{p}_j\|_2^2 \tag{5}$$

where $\mathbf{P}_{y_i}$ is the set of prototype vectors allocated to the class y_i. The separation loss R_{sep} encourages every patch of a *manipulated* training video to stay away from the *real* dynamic prototypes (vice versa), and is given by

$$R_{sep}(\cdot) = -\frac{1}{N} \sum_{i=1}^{N} \min_{\mathbf{p}_j \notin \mathbf{P}_{y_i}} \min_{\mathbf{z} \in \text{patches}(\mathbf{x}_i)} \|\mathbf{z} - \mathbf{p}_j\|_2^2 \tag{6}$$

These loss functions have also been commonly used in previous prototypical-based frameworks [12, 42].

Diversity loss. Due to intra-class prototypes overlapping, we propose a cosine similarity-based regularization term which penalizes prototype vectors of the same class for overlapping with each other, given by

$$R_{div}(\cdot) = \sum_{k=1}^{K} \sum_{\substack{i \neq j \\ \mathbf{p}_i, \mathbf{p}_j \in \mathbf{P}_k}} \max(0, \cos(\mathbf{p}_i, \mathbf{p}_j) - s_{max}) \tag{7}$$

where s_{max} is a hyperparameter for the maximum similarity allowed. This cosine similarity-based loss considers the angle between the prototype vectors regardless of their length. It allows us to penalize the similarity between the prototypes up to a threshold, leading to more diverse and expressive dynamic representations.

3.2.2 Prototype projection and grounding

To achieve grounding, while training we intersperse the following projection step after every few epochs. Specifically, we project the prototype vectors to actual video patches from training videos that contain those dynamics as follows,

$$\mathbf{p}_j \leftarrow \text{argmin}_{\mathbf{z}' \in \text{patches}(f(\mathbf{x}_i))} \|\mathbf{z}' - \mathbf{p}_j\|_2^2 \;\; \forall i \text{ s.t. } y_i = k \tag{8}$$

for all prototype vectors of class k, i.e. $\mathbf{p}_j \in \mathbf{P}_k$. This step projects each prototype vector of a given class to the closest latent representation of a manipulated / genuine video patch that is also from that same class. As a result, the prediction of a test video is made based on the similarities it has with the learned dynamic prototypes. Consequently, the test predictions are *grounded* on the training videos.

3.3. Temporal logic verification via prototypes

Our DPNet architecture allows for the usage of formal methods to verify the compliance of our model. Given the direct computational path from f to p to w, we can check whether the learned prototypes satisfy some desired temporal behaviors. Here, we used Timed Quality Temporal Logic (TQTL) similar to [14], but instead, we consider each video as a *data stream* with each frame as a *time-step*. We hereby give a brief summary of the TQTL language, and the specifications we used to verify our model.

The set of *Timed Quality Temporal Logic* (TQTL) formulas ϕ over a finite set of Boolean-value predicates Q over attributes of prototypes, a finite set of time variables (V_t), and a finite set of prototype indexes (V_p) is inductively defined according to the following grammar:

$$\begin{aligned} \phi ::= \texttt{true} \mid \pi \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \,\mathbf{U}\, \phi_2 \mid \\ x \leq y + n \mid x.\phi \mid \exists p_i @ x, \phi \end{aligned} \tag{9}$$

where $\pi \in Q$, and ϕ_1 and ϕ_2 are valid TQTL formulas. π has the functional form $\pi \equiv F_\pi(t_{1\dots n}, p_{1\dots m}) \sim C$, where

$\sim$ is a comparison operator, i.e. $\sim \in \{<, \leq, >, \geq, =, \neq\}$, and $C \in \mathbb{R}$. For example, the predicate for "the similarity of prototype $\mathbf{p}_1$ to an input at time step 2 is greater than 0.9" is $F(t_2, p_1) > 0.9$. We hereby use $S(\cdot)$ to denote the prototype similarity score.

In the grammar above, $x, y \in V_t, n \in \mathbb{N}, p_i \in V_p$, and **U** is the "until" operator. The time constraints of TQTL are represented in the form of $x \leq y + n$. The freeze time quantifier $x.\phi$ assigns the current time to a variable x before processing the subformula ϕ. The quantifiers $\exists$ and $\forall$ respectively existentially and universally quantify over the prototypes in a given frame. In addition, we use three additional operators: (ψ Implies ϕ) $\psi \rightarrow \phi \equiv \neg\psi \vee \phi$, (Eventually ψ) $\Diamond\psi \equiv \texttt{true}\,\mathbf{U}\,\psi$, and (Always ψ) $\Box\psi \equiv \neg\Diamond\neg\psi$. The semantics of TQTL can be find in the Appendix B.

From the dynamic prototypes, we define *specifications* to check the robustness of our model. We verify that if our model predicts $fake$ for a testing video V, throughout the video, there exists a clip starting at time t where a prototype $\mathbf{p}_i \in \mathbf{P}_{fake}$ is most similar to it compared to all prototypes of the $real$ class $\mathbf{p}_k \in \mathbf{P}_{real}$ for all time t' s.t. $0 \leq t' \leq T_V$. This verifies that there exists of a key region of the video that our prototypes 'see' $fake$ strongly. The formula ϕ_1 denotes this *key-frame* specification, $\mathcal{F}$ = fake, $\mathcal{R}$ = real:

$$\begin{aligned}\phi_1 = \; & \Diamond(t.\exists p_k @t, Class(V) = \mathcal{F}(\mathcal{R}) \wedge p_k \in P_{\mathcal{F}(\mathcal{R})} \\ & \rightarrow \Box(t'.((0 < t' \wedge t' < T_V) \\ & \rightarrow \forall p_j @t', p_j \in P_{\mathcal{R}(\mathcal{F})} \wedge S(t, p_k) > S(t', p_j)\ )))\end{aligned}$$

Moreover, we can specify that if a prototype is *non-relevant*, its similarity to a frame should be consistently low throughout. An example of this safety specification is that a prototype representing a $fake$ temporal artifact should not be highly activated in a $real$ video at any time. We specify this notion below, where numerical values are *user-specified thresholds* that control the strictness of the specification. The formula ϕ_2 denotes this *non-relevance* specification:

$$\begin{aligned}\phi_2 = \; & \Box(t.\forall p_i @t, Class(V) = \mathcal{F}(\mathcal{R}) \wedge p_i \in P_{\mathcal{R}(\mathcal{F})} \\ & \rightarrow S(t, p_i) < 0.4 \wedge \Box(t'.(t \leq t' \wedge t' \leq t + 5) \\ & \rightarrow |S(t', p_i) - S(t, p_i)| < 0.1))\end{aligned}$$

4. Experiments

4.1. Experimental settings

Training datasets. For training our networks, we use the FaceForensics++ (FF++) [49] dataset, which consisted of 1000 original video sequences that have been manipulated with four face manipulation methods: DeepFakes (DF) [20], Face2Face (F2F) [59], FaceSwap (FS) [21], and NeuralTextures (NT) [58]. FF++ provides ground truth manipulation masks showing which part of the face was manipulated. We preprocess the videos by dumping the frames and crops out facial areas based on facial landmarks [72].

Table 1. **Basic information of training and testing datasets.** [36]

Dataset	Real		Fake	
	Video	Frame	Video	Frame
FaceForensics++ (FF++) [49]	1000	509.9k	4000	2039.6k
DeepFakeDetection (DFD) [16]	363	315.4k	3068	2242.7k
DeeperForensics-1.0 [27]	0	0.0k	11000	5608.9k
Celeb-DF [37]	590	225.4k	5639	2116.8k

Testing datasets. To evaluate the cross-dataset generalizability of our aprpoach, we use the following datasets: 1) FaceForensics++ [49] (FF++); 2) DeepfakeDetection (DFD) [16] including 363 real and 3068 deepfake videos released by Google in order to support developing deepfake detection methods; 3) DeeperForensics-1.0 [27] - a large scale dataset consisting of 11, 000 deepfake videos generated with high-quality collected data vary in identities, poses, expressions, emotions, lighting conditions, and 3DMM blendshapes; 4) Celeb-DF [37], a new DeepFake dataset of celebrities with 408 real videos and 795 synthesized video with reduced visual artifacts. (Table 1).

Implementation details. During `DPNet` training phases, the input is formed by stacking 1 RGB frame followed by 10 pre-computed optical flow fields that are uniformly separated. `DPNet` uses a pre-trained HRNet as a backbone network, warm started with ImageNet pre-trained weights. Since our input contains temporal frames, we also perform cross-modality pre-training to average and increase the number of channels in the first conv. layer. The encoder f and prototypes $\mathbf{p}$ were trained with learning rates of $2e^{-4}$ and $1e^{-3}$ respectively. From cross-validation, $\lambda_c, \lambda_s, \lambda_d$ are set to (0.2, -0.2, 0.1), $s_{max} = 0.3$, $m_k = 50$, and prototype vectors are randomly initialized. Further details of the experimental settings are provided in Appendix A.

Evaluation metrics. In addition to the area under the receiver operating curve (AUC), we further use global metrics at a low false alarm rate, similar to [40]. These metrics require critical detectors to operate at a very low false alarm rate, especially in the practical use case of automatic screening for fakes on social media. We used the standardized partial AUC or pAUC in addition to the True Acceptane Rate at low False Acceptance Rate. Lastly, we also inspect the Equal Error Rate, similar to [44].

4.2. Evaluations on Different Quality Settings

In Table 2, we present a thorough comparison of `DPNet` on FF++ [49], as well as baselines that do not have interpretable interpretations or temporal aspects. We train and test with four manipulations types (Deepfakes, FaceSwap, Face2Face, and NeuralTextures) along with the real faces for both High Quality (c23) and Low Quality (c40) compressions. We look at different evaluation metrics such as AUC, pAUC$_{10\%}$, TAR$_{10\%}$, and EER. In general, our approach has superior performance compared to Xception and Two-branch. In particular, we improved AUC from 92% to 99% on HQ (c23) compression, and similarly, AUC

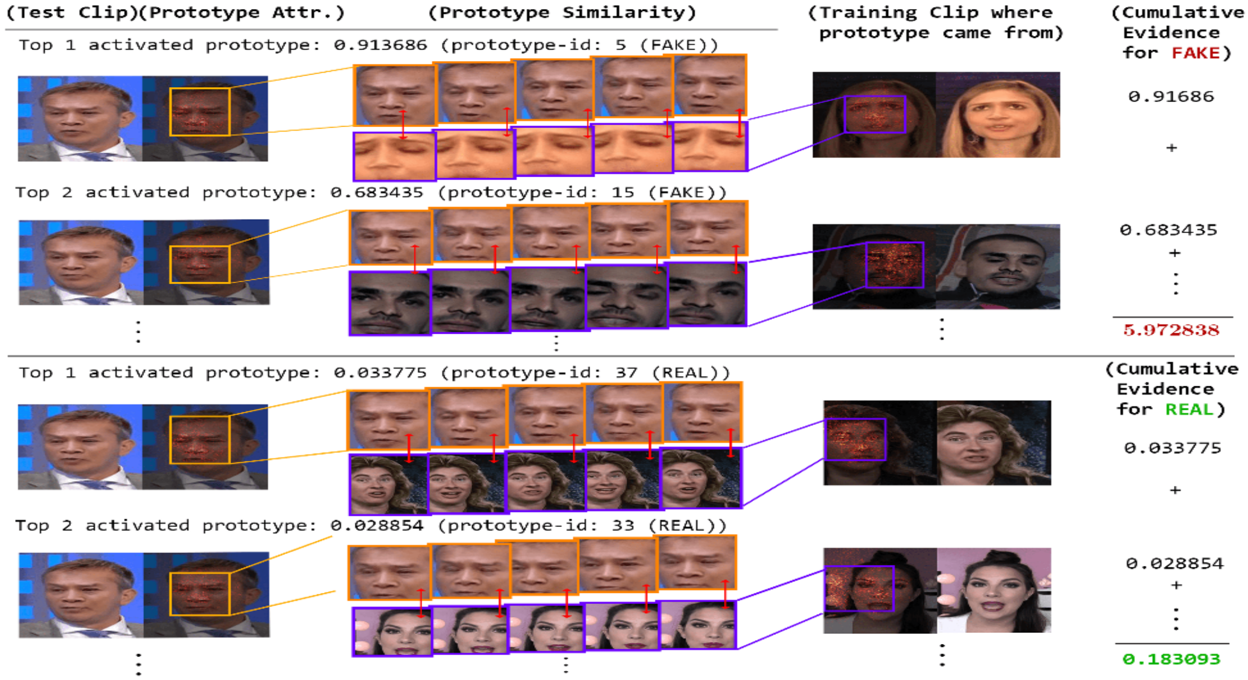

(Test Clip)(Prototype Attr.)
(Prototype Similarity)
(Training Clip where prototype came from)
(Cumulative Evidence for FAKE)
Top 1 activated prototype: 0.913686 (prototype-id: 5 (FAKE))
0.91686
+
Top 2 activated prototype: 0.683435 (prototype-id: 15 (FAKE))
0.683435
+
5.972838
Top 1 activated prototype: 0.033775 (prototype-id: 37 (REAL))
(Cumulative Evidence for REAL)
0.033775
+
Top 2 activated prototype: 0.028854 (prototype-id: 33 (REAL))
0.028854
+
0.183093

Figure 3. **The reasoning process of the DPNet.** The prediction for video clip is based on the similarity comparison between the latent input representations against the learned prototypes. The network tries to find evidence for manipulated/genuine by looking at which spatial/temporal patch that was mostly activated by the dynamic prototypes. Similarity scores between class-specific prototypes are summed.

Table 2. **Comparison on FF++ for different quality settings.** Quantitative results reported for medium compression (c23) and high compression (c40) on FF++ comparing our method with other non-interpretable, interpretable, and temporal methods. Results are reported on four manipulations.

Model	Compression Quality	FF++			
		AUC	$\text{pAUC}_{10\%}$	$\text{TAR}_{10\%}$	EER
DSP-FWA [35]	HQ (c23)	56.89	51.33	14.60	-
Xception [49]		92.30	87.71	81.21	-
Two-branch [40]		98.70	97.43	97.95	-
ProtoPNet [12]		97.95	93.26	94.82	6.00
DPNet (Ours)		**99.20**	**98.21**	**98.04**	3.41
DSP-FWA [35]	LQ (c40)	59.15	52.04	8.82	-
Xception [49]		83.93	74.78	63.25	-
Two-branch [40]		86.59	69.71	62.48	-
ProtoPNet [12]		77.19	61.41	40.18	30.00
DPNet (Ours)		**90.91**	**81.46**	**79.46**	13.35

from 83% to 91% on LQ (c40). The result is also consistent for other low false alarm metrics. Note that Xception and Two-branch does not offer any form of intrinsic interpretability. The table also reports the result of an image-based interpretable method ProtoPNet and a self-supervised method DSP-FWA. Our approach scores the highest AUC across manipulations for both the compression levels, and the usage of temporal information gives `DPNet` an advantage generalizing to more complex and unseen deepfake datasets, seen in the next section.

4.3. Evaluations on Unseen Datasets

Table 3 shows the benchmark results of our framework on the detection of popular unseen deepfake datasets. We evaluate our model's transferability to Google's DeepfakeDetection, DeeperForensics-1.0, and Celeb-DF, given that it is trained only on FaceForensics++ and *zero* external self-supervised data [33]. For fair comparisons, we trained the model on FF++ with all four manipulation types and real faces, and additionally trained an HRNet-only baseline for binary classification. Table 3 reports a clear net improvement over the state-of-the-art, 1-4% in AUC across the board, even for manipulations with real-world perturbations (DeeperForensics) and low visual artifacts (Celeb-DF). Table 4 additionally reports the classic evaluation performance in terms of AUC between our dynamic prototype approach and a wider range of very recent methods on Celeb-DF.

The effect of temporal information. Our network makes predictions about face forgery by exploiting temporal artifacts, which are harder to hide and can generalize across different manipulations in the RGB pixel space. Table 2 indicates general improvement over not only Xception, but also the image-based interpretable ProtoPNet, where the gap is much wider as visual quality degrades. Patch-based comparison in ℓ_2-space looks at the visual patch (enforced by the loss), and significant changes to the video visuals can impact interpretability. This can be further seen in Table 3, where both ProtoPNet and DPNet performs comparably on Celeb-DF, but differ significantly for DeeprForensics. This is not surprising as DeeperForensics features a wider range of distortions, using compressions, color saturations, contrast changes, white gaussian noises, and local block-wise

Table 3. **Generalization ability evaluation on unseen datasets.** Our network trained only on FF++ (c23) performs competitively to state-of-the-art baselines while also providing meaningful interpretations. Temporal artifacts learned by our approach can generalize well to datasets with a wider range of visual distortions such as DeeperForensics (Fig. 5).

Model	Train Set	Unseen Test Set											
		DFD				DeeperForensics				Celeb-DF			
		AUC	$\text{pAUC}_{10\%}$	$\text{TAR}_{10\%}$	EER	AUC	$\text{pAUC}_{10\%}$	$\text{TAR}_{10\%}$	EER	AUC	$\text{pAUC}_{10\%}$	$\text{TAR}_{10\%}$	EER
Xception [49]	FF++	91.27	81.25	**78.97**	16.61	87.89	77.84	65.42	19.90	63.93	55.00	21.47	39.89
HRNet [65]		89.35	79.73	71.35	18.34	83.57	74.75	62.92	22.64	61.22	54.68	19.27	45.88
ProtoPNet [12]		84.46	77.58	66.42	23.81	61.59	52.26	14.65	38.81	**69.33**	**56.06**	**29.12**	**36.52**
DPNet (Ours)		**92.44**	**81.30**	76.21	**16.21**	**90.80**	**79.66**	**75.67**	**17.30**	68.20	55.02	25.88	37.08

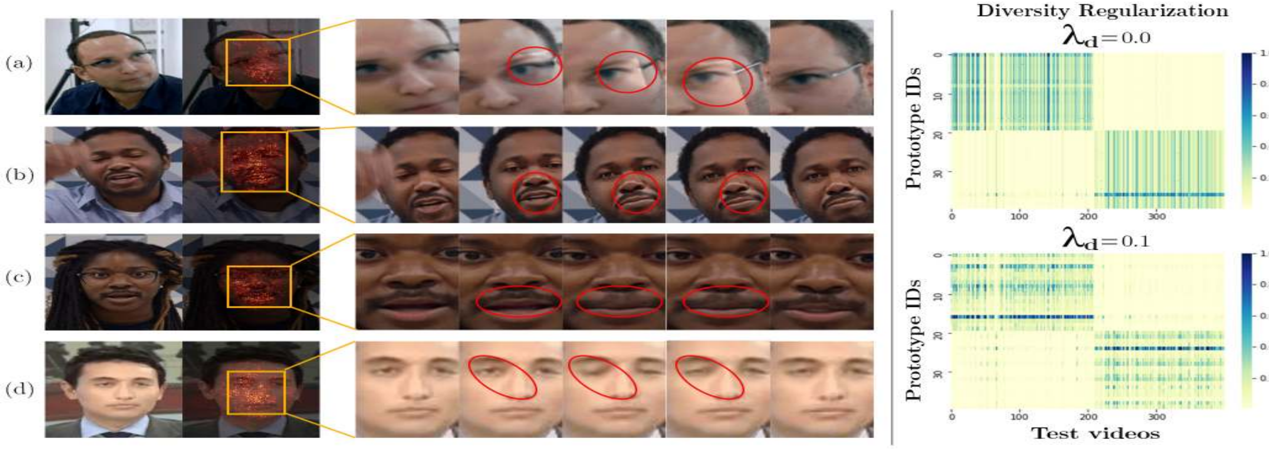


Figure 4. **Visualization of learned dynamic prototypes.** Left column depicts the different classes of temporal artifacts and unnatural movements found by `DPNet`. Right column shows the effect of diversity regularization (7) on prototype similarities across test videos.

distortions [27] (Figure 5). And since the number of prototypes are fixed, high performance requires generalizable prototypes. Table 5 further investigates the impact of varying the number of flow frames.

Qualitative visualization of learned dynamic prototypes. Figure 4 presents classes of temporal artifacts captured by the learned dynamic prototypes. We take the gradient attribution with respect to each prototype similarity score. During the training steps, while projecting, we kept track of the latent patch within the original training clip that is closest to each prototype. Hence, during testing, we can visualize the prototypes that are most similar to the testing videos. Some artifacts features heavy discoloration (Fig. 4a), which is already interpretable as images, but changes in discoloration over time offer more interpretability. In fact, other artifacts such as subtle discolorations or movements (Fig. 4b,c) are much harder to interpret with just one image, especially the facial jitterings and unnatural oscillations (Fig. 4d).

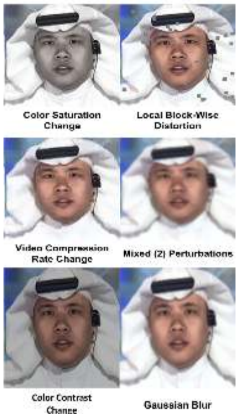
4.4. Temporal specifications over prototypes

Table 6 reports the robustness of `DPNet` and interpretable baselines against desired temporal specifications specified in Section 3.3. Overall, both approaches satisfy the ϕ_{1,key_frame} specification up to high-percentage, with `DPNet` performing better, especially with the $fake$ traces. This indicates that with high-probability, we can find a key-frame within the video that is most relevant in explaining

Table 4. **Best competing methods on Celeb-DF are reported.** Results for other methods are from [37]

Model	FF++	Celeb-DF
MesoInception4 [1]	83.0	53.6
Two-stream [74]	70.1	53.8
HeadPose [70]	47.3	54.6
VA-MLP [41]	66.4	55.0
VA-LogReg	78.0	55.1
Xception-raw [49]	**99.7**	48.2
Xception-c23	**99.7**	65.3
Xception-c40	95.5	65.5
Multi-task [44]	76.3	54.3
Capsule [45]	96.6	57.5
DSP-FWA [35]	93.0	64.6
ProtoPNet [12]	98.0	69.3
DPNet (Ours) - c23	99.2	68.2
DPNet (Ours) - c40	90.91	**71.76**

Figure 5. **Visual distortions in DeeperForensics** [27]. Not in FF++.



why a video is a deepfake, via a dynamic prototypes in the $fake$ class. On the other hand, the models perform poorly with the stringent specification $\phi_{2,non\_relevance}$, which requires non-relevant prototypes to both stay low and not change at all over time. Since `DPNet` utilized temporal information via optical flows, this is a stricter specification to enforce as flows can change drastically across consecutive frames, hence the lower percentage of satisfying traces. We experiment by relaxing the consecutive "next 5 frames" non-changing constraint, $\phi_{3,relaxed}$, which now only enforce non-relevant prototype similarities to be low. The flex-

Table 5. **Ablation study on FF++.** (a) Testing metrics obtained by training varying the number of prototypes m_k under (c23) compression, testing on unseen DFD. (b) Ablation experiments showing the impact of length of the input flow segments, testing on unseen DeeperForensics (c) Ablation experiments on (c23) compression without diversity regularization.

(a)	FF++ - DFD - varying num prototypes			
	AUC	$\text{pAUC}_{10\%}$	$\text{TAR}_{10\%}$	EER
$m_k = 10$	86.19	74.09	56.13	21.93
$m_k = 25$	90.79	77.36	71.38	17.46
$m_k = 50$	92.44	81.30	76.21	16.21
$m_k = 100$	91.88	79.94	77.70	17.1

(b)	FF++ - DeeperForensics - wo/ flow frames			
	AUC	$\text{pAUC}_{10\%}$	$\text{TAR}_{10\%}$	Time per Batch
`- flows`	61.59	52.26	14.65	0.68 ± 0.13s
`frame` = 5	88.03	77.76	69.38	1.11 ± 0.13s
`frame` = 10	90.80	79.66	75.67	2.22 ± 0.44s
`frame` = 15	84.85	75.22	66.98	4.39 ± 0.34s

(c)	FF++ c23 - wo/ Diversity		
	AUC	$\text{pAUC}_{10\%}$	$\text{TAR}_{10\%}$
`- diversity`	97.81	96.17	95.76
DPNet	99.20	98.21	98.04

Table 6. **Percentage of traces satisfying temporal specifications.** Rows in each block represent the percentage over positive, negative, and all traces.

		ϕ_{1,key_frame}	$\phi_{2,non\_relevance}$	$\phi_{3,relaxed}$
ProtoPNet	(+)	95.23	49.73	70.89
	(-)	89.09	42.18	58.76
		92.00	45.75	64.50
DPNet (Ours)	(+)	93.65	28.04	74.07
	(-)	91.46	16.11	56.39
		92.50	21.75	64.75

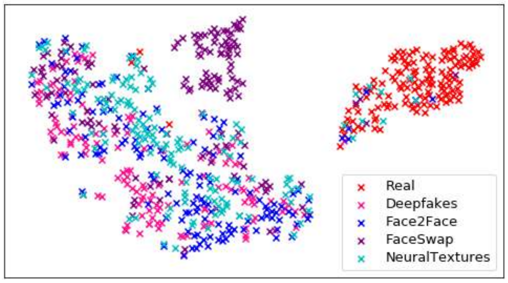


Figure 6. **The t-SNE embedding visualization of the DPNet trained for binary classification on FF++ (c23)**. Red X's are real videos, others represent data generated by different manipulations.

ibility of temporal logic along with interpretable model allows end-users to specify and enforce certain desiderata in their detection framework. This further increases trustworthiness and interpretability of these detection frameworks.

4.5. Ablation study

Choosing the number of prototypes m_k. Table 5a shows the ablation experiments when choosing the number of prototypes. We further investigate the generalization impact of this decision on unseen deepfakes (DFD). Using the same hyperparameter configuration, the parameter is set $m_k = \{10, 25, 50, 100\}$. As shown in Table 5a , the AUC first improves dramatically as m_k increases, but continuing to do so yields diminishing returns. Thus, there is an inherent trade-off between predictive performance and interpretability. In practice, since increasing k after a certain threshold only brings marginal improvement to the performance, we would suggest to gradually increase m_k, until a desired low false alarm threshold is achieved.

Ablation study on the number of flow frames. Table 5b further reports the generalization performance to DeeperForensics when varying the number of flow frames utilized by the network. Here we test the generalization of the dynamic prototypes and how the length of the time segment given to the network impacts its performances. As shown in Figure 5, visual elements of deepfakes can change drastically across domains and manipulations. However, temporal information can be key in pinpointing artifacts shared by all manipulations. Using the same configuration as previous, the number of flow frames is set `frame` $= \{0, 5, 10, 15\}$, where zero frames do not use flows, similar to the ProtoPNet baseline. In general, Table 5b reports a significant net improvement on the unseen dataset DeeperForensics, where there are a larger number of visual distortions. In addition, we observe that using a small number of flow frame can already yield a better performance, and that using a longer sequence can negatively impact prediction as the inputs become noisier. Moreover, the trade-off between predictive performance and training time will also have to be considered since using more frames significantly impact each batch loading and processing time.

The effect of diversity regularization λ_d. To study the effect of the diversity regularization term, we removed the term by setting $\lambda_d = 0$ measure the AUC drop on FF++. Results are shown in Table 6c. We further examine the impact of λ_d by plotting the similarity scores between the prototypes and test videos as heatmaps (Figure 4-right). Without the diversity regularization, most of the rows have similar patterns, indicating that the prototypes are duplicating and are less diverse, yielding lower test AUC on FF++.

5. Conclusion

There is a growing need to use automatic deepfake detection models to detect and combat deepfakes. However, a reliance on deepfake detectors places enormous trust on these models. This work aims to help justify this trust by improving the interpretability of deepfake detection. In addition to model interpretability, this work offers insights into what parts of deepfake videos can be used to discern deepfakes, which may inform people how to detect deepfakes themselves. Model interpretations strengthen the accountability of deepfake detectors and our work encourages future research in explaining deepfake detectors.

Acknowledgment

This work is supported by the National Science Foundation under Grant No. 1619458, the FMitF program, Grant No. CCF-1837131, and the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00111990059.

References

[1] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: a compact facial video forgery detection network. *CoRR*, abs/1809.00888, 2018.

[2] Shruti Agarwal, Hany Farid, Yuming Gu, Mingming He, Koki Nagano, and Hao Li. Protecting world leaders against deep fakes. In *CVPR Workshops*, 2019.

[3] David Alvarez-Melis and Tommi S. Jaakkola. Towards robust interpretability with self-explaining neural networks. *CoRR*, abs/1806.07538, 2018.

[4] Irene Amerini, Leonardo Galteri, Roberto Caldelli, and Alberto Bimbo. Deepfake video detection through optical flow based cnn. In *CVPR Workshops*, pages 1205–1207, 10 2019.

[5] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10(7):e0130140, 07 2015.

[6] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L. Morency. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, pages 59–66, 2018.

[7] Tarek R. Besold, Artur S. d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro M. Domingos, Pascal Hitzler, Kai-Uwe Kühnberger, Luís C. Lamb, Daniel Lowd, Priscila Machado Vieira Lima, Leo de Penning, Gadi Pinkas, Hoifung Poon, and Gerson Zaverucha. Neural-symbolic learning and reasoning: A survey and interpretation. *ArXiv*, abs/1711.03902, 2017.

[8] Patricia Bouyer. Model-checking timed temporal logics. *Electronic Notes in Theoretical Computer Science*, 231:323 – 341, 2009. Proceedings of the 5th Workshop on Methods for Modalities (M4M5 2007).

[9] Sarah Cahlan. How misinformation helped spark an attempted coup in gabon. *The Washington Post*, 2020.

[10] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[11] Lucy Chai, David Bau, Ser-Nam Lim, and Phillip Isola. What makes fake images detectable? understanding properties that generalize. *arXiv preprint arXiv:2008.10588*, 2020.

[12] Chaofan Chen, Oscar Li, Alina Barnett, Jonathan Su, and Cynthia Rudin. This looks like that: deep learning for interpretable image recognition. *CoRR*, abs/1806.10574, 2018.

[13] Davide Cozzolino, Justus Thies, Andreas Rössler, Christian Riess, Matthias Nießner, and Luisa Verdoliva. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv preprint arXiv:1812.02510*, 2018.

[14] Adel Dokhanchi, Hani Ben Amor, Jyotirmoy V. Deshmukh, and Georgios Fainekos. Evaluating perception systems for autonomous vehicles using quality temporal logic. In Martin Leucker and Christian Colombo, editors, *Runtime Verification- 18th International Conference, RV 2018, Proceedings*, pages 409–416. Springer Verlag, Jan. 2019.

[15] Mengnan Du, Shiva Pentyala, Yuening Li, and Xia Hu. Towards generalizable forgery detection with locality-aware autoencoder. *arXiv preprint arXiv:1909.05999*, 2019.

[16] Nicholas Dufour, Andrew Gully, Per Karlsson, Alexey Victor Vorbyov, Thomas Leung, Jeremiah Childs, and Christoph Bregler. Deepfakes detection dataset by google & jigsaw.

[17] Matthew B. Dwyer, George S. Avrunin, and James C. Corbett. Property specification patterns for finite-state verification. In *Proceedings of the Second Workshop on Formal Methods in Software Practice*, page 7–15, New York, NY, USA, 1998. Association for Computing Machinery.

[18] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. *CoRR*, abs/1812.03982, 2018.

[19] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3429–3437, 2017.

[20] Deepfakes github. https://github.com/deepfakes/faceswap. Accessed on 2018-10-29.

[21] Deepfakes github. https://github.com/MarekKowalski/FaceSwap/. Accessed on 2018-10-29.

[22] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2015.

[23] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy A. Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *ArXiv*, abs/1810.12715, 2018.

[24] Karen Hao. An ai app that "undressed" women shows how deepfakes harm the most vulnerable. *MIT Technology Review*, 2019.

[25] Xiaowei Huang, Marta Z. Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *CAV*, 2017.

[26] David Ingram. A face-swapping app takes off in china, making ai-powered deepfakes for everyone. *NBC*, 2019.

[27] Liming Jiang, Ren Li, Wayne Wu, Chen Qian, and Chen Change Loy. Deeperforensics-1.0: A large-scale dataset for real-world face forgery detection. In *CVPR*, 2020.

[28] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. *ArXiv*, abs/1702.01135, 2017.

[29] A. Khodabakhsh, R. Ramachandra, K. Raja, P. Wasnik, and C. Busch. Fake face detection methods: Can they be generalized? In *2018 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–6, 2018.

[30] Been Kim, Cynthia Rudin, and Julie A Shah. The bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Advances in Neural Information Processing Systems*, pages 1952–1960, 2014.

[31] Pavel Korshunov and Sébastien Marcel. Deepfakes: a new threat to face recognition? assessment and detection. *ArXiv*, abs/1812.08685, 2018.

[32] Mathias Lécuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adver-

sarial examples with differential privacy. *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672, 2018.

[33] Lingzhi Li, Jianmin Bao, Ting Zhang, Hao Yang, Dong Chen, Fang Wen, and Baining Guo. Face x-ray for more general face forgery detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5001–5010, 2020.

[34] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. *CoRR*, abs/1710.04806, 2017.

[35] Yuezun Li and Siwei Lyu. Exposing deepfake videos by detecting face warping artifacts. *CoRR*, abs/1811.00656, 2018.

[36] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. *arXiv: Cryptography and Security*, 2019.

[37] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3207–3216, 2020.

[38] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[39] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083, 2017.

[40] Iacopo Masi, Aditya Killekar, Royston Marian Mascarenhas, Shenoy Pratik Gurudatt, and Wael AbdAlmageed. Two-branch recurrent network for isolating deepfakes in videos. *arXiv preprint arXiv:2008.03412*, 2020.

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

[42] Yao Ming, Panpan Xu, Huamin Qu, and Liu Ren. Interpretable and steerable sequence learning via prototypes. *CoRR*, abs/1907.09728, 2019.

[43] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *CoRR*, abs/1706.07979, 2017.

[44] Huy H. Nguyen, Fuming Fang, Junichi Yamagishi, and Isao Echizen. Multi-task learning for detecting and segmenting manipulated facial images and videos. *CoRR*, abs/1906.06876, 2019.

[45] Huy H. Nguyen, Junichi Yamagishi, and Isao Echizen. Capsule-forensics: Using capsule networks to detect forged images and videos. *CoRR*, abs/1810.11215, 2018.

[46] Oded Padon, Jochen Hoenicke, Giuliano Losa, Andreas Podelski, Mooly Sagiv, and Sharon Shoham. Reducing liveness to safety in first-order logic. *Proc. ACM Program. Lang.*, 2(POPL), Dec. 2017.

[47] Ivan Petrov, Daiheng Gao, Nikolay Chervoniy, Kunlin Liu, Sugasa Marangonda, Chris Um'e, Mr. Dpfks, RP Luis, Jian Jiang, Sheng Zhang, Pingyu Wu, Bo Zhou, and Weiming Zhang. Deepfacelab: A simple, flexible and extensible face swapping framework. *ArXiv*, abs/2005.05535, 2020.

[48] Yuyang Qian, Guojun Yin, Lu Sheng, Zixuan Chen, and Jing Shao. Thinking in frequency: Face forgery detection by mining frequency-aware clues. *arXiv preprint arXiv:2007.09355*, 2020.

[49] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. *CoRR*, abs/1901.08971, 2019.

[50] Ekraam Sabir, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, and Prem Natarajan. Recurrent convolutional strategies for face manipulation detection in videos. *CoRR*, abs/1905.00582, 2019.

[51] Motoki Sato, Jun Suzuki, Hiroyuki Shindo, and Yuji Matsumoto. Interpretable adversarial perturbation in input embedding space for text. *arXiv preprint arXiv:1805.02917*, 2018.

[52] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016.

[53] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. *CoRR*, abs/1704.02685, 2017.

[54] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *preprint*, 12 2013.

[55] A Prasad Sistla. Safety, liveness and fairness in temporal logic. *Formal Aspects of Computing*, 6(5):495–511, 1994.

[56] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *CoRR*, abs/1703.01365, 2017.

[57] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2014.

[58] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[59] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. *Commun. ACM*, 62(1):96–104, Dec. 2018.

[60] Rob Toews. Deepfakes are going to wreak havoc on society. we are not prepared. *Forbes*, 2020.

[61] Rubén Tolosana, Rubén Vera-Rodríguez, Julian Fiérrez, Aythami Morales, and Javier Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection. *ArXiv*, abs/2001.00179, 2020.

[62] William Turton and Andrew Martin. How deepfakes make disinformation more real than ever. *Bloomberg News*, 2020.

[63] Cristian Vaccari and Andrew Chadwick. Deepfakes and disinformation: Exploring the impact of synthetic political video on deception, uncertainty, and trust in news. *Social Media + Society*, 6(1):2056305120903408, 2020.

[64] Marcell Vazquez-Chanlatte, Shromona Ghosh, Jyotirmoy V. Deshmukh, Alberto L. Sangiovanni-Vincentelli, and Sanjit A. Seshia. Time series learning using monotonic logical properties. *CoRR*, abs/1802.08924, 2018.

[65] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[66] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. *CoRR*, abs/1608.00859, 2016.

[67] Run Wang, Felix Juefei-Xu, Lei Ma, Xiaofei Xie, Yihao Huang, Jian Wang, and Yang Liu. Fakespotter: A simple yet robust baseline for spotting ai-synthesized fake faces. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2020.

[68] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. In *Advances in Neural Information Processing Systems*, pages 6367–6377, 2018.

[69] Lily Weng, Pin-Yu Chen, Lam Nguyen, Mark Squillante, Akhilan Boopathy, Ivan Oseledets, and Luca Daniel. PROVEN: Verifying robustness of neural networks with a probabilistic approach. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6727–6736, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[70] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses. *CoRR*, abs/1811.00661, 2018.

[71] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.

[72] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[73] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. *European Conference on Computer Vision*, 2018.

[74] Peng Zhou, Xintong Han, Vlad I. Morariu, and Larry S. Davis. Two-stream neural networks for tampered face detection. *CoRR*, abs/1803.11276, 2018.

[75] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017.