Mumpy: Multilateral Temporal-view Pyramid Transformer for Video Inpainting Detection

Ying Zhang[1]
yingzhang@stu.ouc.edu.cn
Yuezun Li[1,✉]
liyuezun@ouc.edu.cn
Bo Peng[2]
bo.peng@nlpr.ia.ac.cn
Jiaran Zhou[1]
zhoujiaran@ouc.edu.cn
Huiyu Zhou[3]
hz143@leicester.ac.uk
Junyu Dong[1]
dongjunyu@ouc.edu.cn

[1] School of Computer Science and Technology,
Ocean University of China
[2] New Laboratory of Pattern Recognition (NLPR),
Institute of Automation, Chinese Academy of Sciences (CASIA)
[3] School of Computing and Mathematical Sciences,
University of Leicester

✉: Corresponding author

Abstract

The task of video inpainting detection is to expose the pixel-level inpainted regions within a video sequence. Existing methods usually focus on leveraging spatial and temporal inconsistencies. However, these methods typically employ fixed operations to combine spatial and temporal clues, limiting their applicability in different scenarios. In this paper, we introduce a novel Multilateral Temporal-view Pyramid Transformer (*MumPy*) that collaborates spatial-temporal clues flexibly. Our method utilizes a newly designed multilateral temporal-view encoder to extract various collaborations of spatial-temporal clues and introduces a deformable window-based temporal-view interaction module to enhance the diversity of these collaborations. Subsequently, we develop a multi-pyramid decoder to aggregate the various types of features and generate detection maps. By adjusting the contribution strength of spatial and temporal clues, our method can effectively identify inpainted regions. We validate our method on existing datasets and also introduce a new challenging and large-scale Video Inpainting dataset based on the YouTube-VOS dataset, which employs several more recent inpainting methods. The results demonstrate the superiority of our method in both in-domain and cross-domain evaluation scenarios.

1 Introduction

Video inpainting is an emerging technique that aims to recover the disrupted or designated regions in sequences while maintaining consistent semantic context [12, 13, 18, 21, 27, 41]. In recent years, there has been remarkable progress in this technique with the ever-growing

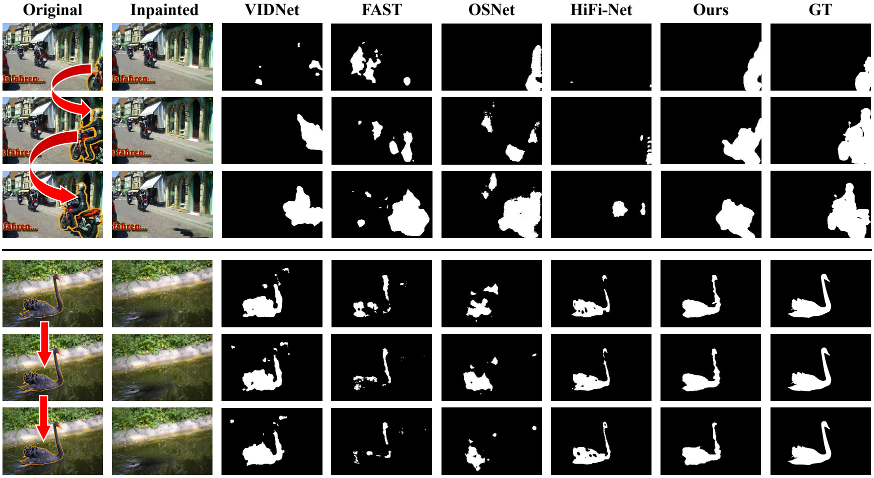

Figure 1: Results of our method compared with the others in cross-domain scenarios. The top examples show an obvious temporal relationship while the bottom ones exhibit a strong spatial relationship. These examples demonstrate the significance of flexible collaboration of spatial-temporal clues.

deep generative methods. As a result, the realism of inpainted visuals has significantly improved, making it increasingly challenging for human observers to detect the manipulations. The misuse of this technique can lead to serious concerns, such as removing crucial evidence or objects to deceive judges or mislead public opinion [5, 14, 28]. Therefore, detecting video inpainting has become an urgent need in digital forensics, especially with the rapid advancement of large-scale vision generative models [7, 10, 35].

In recent years, many methods have been proposed for video inpainting detection [30, 36, 40]. One typical solution to detect the inpainted regions is by leveraging the frame-level spatial inconsistency clues, *e.g.*, [14, 15, 16]. Due to the nature of the video sequence, inpainted regions can also introduce temporal inconsistency. Thus, more recent methods incorporate sequential-based strategies to explore both spatial and temporal inconsistencies [30, 36, 40]. Typically, these methods employ deep models to extract spatial and temporal features using fixed operations (*e.g.*, convolution with fixed kernel size) for all video sequences, leading to a straightforward combination of these features. However, the importance of spatial or temporal clues may differ across various scenarios. For example, in videos having a strong temporal relationship (*e.g.*, target objects have obvious movement in sequence), the inpainted regions are more likely to disrupt temporal consistency. In such cases, paying more attention to temporal clues allows one to identify the manipulation. On the other hand, in videos with strong spatial relationships (*e.g.*, target objects are significantly salient), spatial inconsistency can more effectively identify the manipulated regions. Thus, a more feasible collaboration of spatial-temporal clues is needed in practical applications (see Fig. 1).

In this paper, we describe a *Mu*ltilateral Te*m*poral-view *Py*ramid Transformer (*MumPy*) to expose inpainted regions by adopting a variety of collaboration ways for spatial-temporal clues (see Fig. 2). Our method considers different degrees of importance for spatial and temporal clues and adjusts the contribution strength of each clue accordingly. To achieve this, we develop a Multilateral Temporal-view Encoder (Sec. 3.1), a Deformable Window-based Temporal-view Interaction module (Sec. 3.2), and a Multi-pyramid Decoder (Sec. 3.3), respectively. The Multilateral Temporal-view Encoder consists of several branches, each representing a specific collaboration of spatial-temporal clues within different temporal views. To

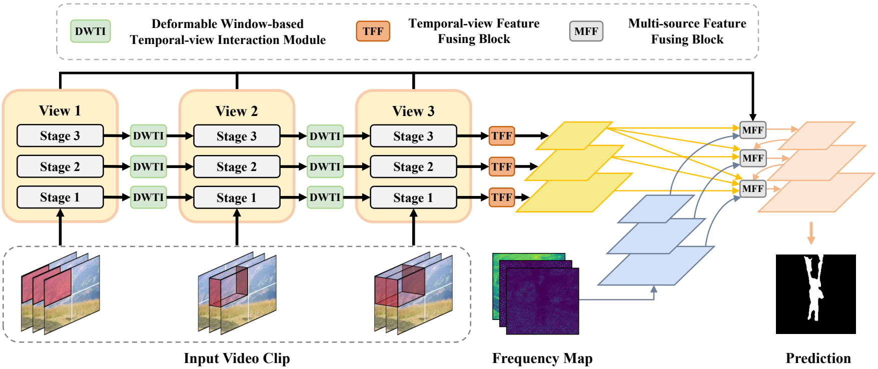
DWTI
Deformable Window-based Temporal-view Interaction Module
TFF
Temporal-view Feature Fusing Block
MFF
Multi-source Feature Fusing Block
View 1
View 2
View 3
Stage 3
Stage 2
Stage 1
DWTI
TFF
MFF
Input Video Clip
Frequency Map
Prediction

Figure 2: Overview of the proposed Multilateral Temporal-view Pyramid Transformer. See text for details.

increase the diversity of collaborations, we propose a deformable window-based temporal-view interaction strategy that fuses the knowledge from adjacent branches. The motivation is a single branch can represent a more comprehensive temporal view if it absorbs the knowledge from others. This strategy adaptively builds the correlation between inpainted regions from different temporal views using deformable-style attention and performs the interaction separately inside windows. Furthermore, we propose a multi-pyramid decoder to generate the detection maps. This decoder makes full use of the intermediate features from the encoder, the frequency signals, and the fusing of high-level features in multiple pyramid schemes.

Following previous works [30, 36, 40], we validate our method on the DAVIS Video Inpainting dataset (DVI) and Free-from Video Inpainting dataset (FVI), demonstrating its efficacy in both in-domain and cross-domain evaluation scenarios. To provide a more comprehensive evaluation, we present a new challenging and large-scale video inpainting dataset based on YouTube-VOS [33]. Compared to DVI (150 videos) and FVI (100 videos), this dataset contains more videos (3471 in total) covering diverse real-world scenarios and incorporates four additional state-of-the-art video inpainting methods (Sec. 4). Our method is further evaluated on this dataset following the same evaluation scenarios, showing its superiority compared to recent counterparts.

Our contributions can be summarized into three-fold: **1)** We describe a new network, *MumPy*, that allows for flexible exploration of spatial-temporal correlations, and propose a novel temporal-view interaction strategy to enhance the diversity of the collaborations between spatial and temporal clues. **2)** We introduce a multi-pyramid decoder that effectively leverages knowledge from various sources, including the accumulation of features from the encoder, the assistance of frequency signals, and the guidance of high-level features. **3)** We present a challenging and large-scale YouTube-VOS Video Inpainting dataset (YTVI), which includes many state-of-the-art video inpainting methods. Extensive experiments on the YTVI dataset, as well as the DVI and FVI datasets, demonstrate the superior performance of our method in both in-domain and cross-domain evaluation.

2 Related Work

Video Inpainting. The improvement of deep generative techniques has greatly made video inpainting effortless and realistic. Notably, recent inpainting methods such as VI [12], OP [21], and CP [13] have drawn a lot of attention. They are involved in DVI dataset due to their favorable performance. More recently, several methods have been proposed to further improve the quality of inpainting using more advanced deep models. For instance, ISVI [38] employs optical flows for inpainting propagation, while EG2 [17] performs propagation in feature space. FF [18] introduces a Transformer model to explore fine-grained information for inpainting, and PP [41] incorporates optical flow with Transformers for inpainting.

Video Inpainting Detection. To detect inpainting videos, many methods have been proposed, *e.g.*, [14, 15, 16, 36, 40]. One direction of the methods focuses on capturing the frame-level inpainting clues. For instance, [15] proposes high-pass pre-filtering to acquire high-frequency residual information, assisting in locating inpainted regions. [14] exploits existing noise discrepancies between authentic and inpainted images. Since these methods focus on the spatial traces at the frame level, they could not utilize the temporal information exhibited in inpainted videos, which possibly degrades their performance in real-world scenarios with prominent object motion.

To leverage the temporal information, there are several methods proposed to take video clips as input, *e.g.*, [30, 36, 40]. The works of [30, 40] extract rich spatial features and utilize LSTM-related structures to extract temporal features. The recent Video Vision Transformers greatly improve the ability to model spatial-temporal features for classification [2, 3, 22, 34]. Inspired by these architectures, the work of [36] introduces spatial-temporal patches obtained from extracted video clips into a vision transformer to model spatial-temporal correlations among patches. However, the existing methods have fixed collaborations between spatial and temporal features, making them less adaptable to various real-world scenarios. Therefore, this paper introduces a flexible approach that enables multiple temporal-view collaborations with dedicated architecture design.

3 Method

3.1 Multilateral Temporal-view Encoder

Denote a video clip as a set of frames $\mathcal{V} = [\mathcal{I}_1, ..., \mathcal{I}_T]$, where $\mathcal{I}_i \in \mathbb{R}^{H\times W\times C}$ denotes the i-th frame. The general video-based Transformers first divide it into several spatial and temporal patches, known as tubelets. Denote these tubelets as $\{b_1, ..., b_N\}$, where $b_i \in \mathbb{R}^{t\times h\times w\times c}$ and $N = \lfloor \frac{T}{t} \rfloor \times \lfloor \frac{H}{h} \rfloor \times \lfloor \frac{W}{w} \rfloor$. Note that in existing methods, the length of tubelets t is typically fixed, indicating a fixed collaboration of spatial and temporal clues. In our method, we consider various collaboration ways of these two clues, inspired by the multimodal spirits [6, 11, 34]. Since t determines the length of the temporal window, different values correspond to different temporal views, and these views indicate various collaborations of spatial and temporal clues. For example, $t = T$ denotes to take the whole sequence into account, while $t = 1$ is degraded to only use spatial patches. Thus we transform the input sequence into multiple types of tokens according to different t. Denote the set of temporal-view (tubelets length) as $\{t_1, ..., t_m\}$, where $t_i \leq t_{i+1}, t_i \geq 1, t_{i+1} \leq T$ and m is the number of temporal-view. Given the video clip $\mathcal{V}$, we employ 3D convolution as in [2] for tokenization. For a temporal view $t \in \{t_1, ..., t_m\}$, the convolution kernel is set to the size of $t \times h \times w$ with stride $t \times h \times w$. Denote the 3D convolution operations at temporal-view t as $\mathcal{T}^{(t)}$. The corresponding tokens from the video clip can be defined as $\mathbf{z}^{0,(t)} = [z_i^{(t)}, ..., z_N^{(t)}]$, where $z_i^{(t)} = \mathcal{T}^{(t)}(b_i)$. Thus we obtain a set of input tokens according to different temporal views as $\{\mathbf{z}^{0,(t_i)}, ..., \mathbf{z}^{0,(t_m)}\}$.

For each temporal view, we build a k-stage Transformer network, and each stage contains several pairs of Swin Transformer blocks [19]. The operation of consecutive Swin

Transformer blocks can be defined as

$$\begin{aligned}&\hat{\mathbf{z}}^{l,(t)} = \text{W-MSA}\left(\text{LN}(\mathbf{z}^{l-1,(t)})\right) + \mathbf{z}^{l-1,(t)},\ \mathbf{z}^{l,(t)} = \text{MLP}\left(\text{LN}(\hat{\mathbf{z}}^{l,(t)})\right) + \hat{\mathbf{z}}^{l,(t)},\\ &\hat{\mathbf{z}}^{l+1,(t)} = \text{SW-MSA}\left(\text{LN}(\mathbf{z}^{l,(t)})\right) + \mathbf{z}^{l,(t)},\ \mathbf{z}^{l+1,(t)} = \text{MLP}\left(\text{LN}(\hat{\mathbf{z}}^{l+1,(t)})\right) + \hat{\mathbf{z}}^{l+1,(t)},\end{aligned} \tag{1}$$

where W-MSA and SW-MSA denote window-based multi-head self-attention and shifted window configuration, LN is layer normalization, MLP is multi-layer perception blocks and l is the index of the Transformer block, respectively. Note that $l = 0$ denotes the input tokenization. After this, we obtain a set of tokens corresponding to different temporal views and merge these tokens as the input of Multi-source Feature Fusing Pyramid (see Sec. 3.3).

3.2 Deformable Window-based Temporal-view Interaction

Ideally, each view can interact with all other views. Yet, this will incur large computational costs. Hence, we opt to perform interaction solely among adjacent branches, following an ascending order of temporal-view. The rationale behind this is that larger views concentrate more on temporal relationships, while smaller views offer richer spatial relationships. Interacting in this way can increase the diversity of larger views. The interaction is accomplished by the cross-attention mechanism. Note that conventional cross-attention operations build the correlations across all feature elements, which may overlook the importance of inpainted regions. To solve this, we propose Deformable Window-based Temporal-view Interaction (DWTI), which employ the deformable attention mechanism [32] to adaptively concentrate on the inpainted regions. Fig. 3 illustrates an overview of this process. Denote two intermediate tokens from adjacent temporal views as $\mathbf{z}^{(t)}$ and $\mathbf{z}^{(t+1)}$[1], where the former comes from a smaller view and the other from a larger view. We convert these two tokens to the same shape of $h' \times w' \times c'$, *i.e.*, $\mathbf{z}^{(t)} \in \mathbb{R}^{h' \times w' \times c'}, \mathbf{z}^{(t+1)} \in \mathbb{R}^{h' \times w' \times c'}$. We generate queries from $\mathbf{z}^{(t+1)}$ and keys, values from $\mathbf{z}^{(t)}$. The queries q are obtained by linearly projecting the tokens $\mathbf{z}_i^{(t+1)}$ as $q = \mathbf{z}_i^{(t+1)} W_q$. Denote p as the uniform grid of points. To obtain the offset for p, we utilize a DNN which can transform the queries q into offset values as $\triangle p = \theta\left(q\right)$. We can obtain the deformed points by integrating reference points p and corresponding offsets $\triangle p$, and sample the features at the locations of deformed points in $\mathbf{z}^{(t)}$ to project to the key and value tokens, defined as

$$\begin{aligned}&q = \mathbf{z}^{(t+1)} W_q, k = \tilde{\mathbf{z}}^{(t)} W_k, v = \tilde{\mathbf{z}}^{(t)} W_v,\\ &\triangle p = \theta\left(q\right), \tilde{\mathbf{z}}^{(t)} = \delta(\mathbf{z}^{(t)}; p + \triangle p),\end{aligned} \tag{2}$$

where $\delta(\cdot)$ is a sampling function. Then we perform MSA by using the obtained q, k, v respectively as $h = \text{Softmax}(qk^{\top}/\sqrt{d})v$.

Considering that the global attention is easy to be disrupted by outliers and requires large computational costs, we describe a window-based method by only considering the cross-attention inside corresponding windows between two temporal views. Note that each token is an integration of K windows, defined as $\mathbf{z}^{(t)} = [\mathbf{z}_1^{(t)}, \ldots, \mathbf{z}_K^{(t)}]$. For each window, we perform the deformable cross-attention using Eq. (2) and then ensemble up the results.

3.3 Multi-pyramid Decoder

We design a CNN-based decoder architecture to take in the features from the encoder and then generate the detection maps. Inspired by the pyramid spirits in computer vision tasks

[1] We omit the index of Transformer block l for simplicity.

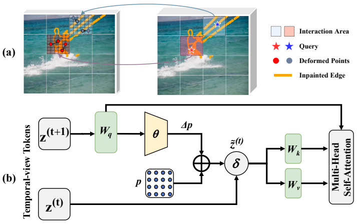

Figure 3: (a) Diagram and (b) process of DWTI.

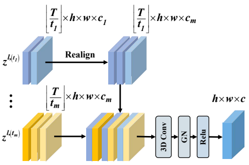
$\lfloor \frac{T}{t_1} \rfloor \times h \times w \times c_1$
$\lfloor \frac{T}{t_1} \rfloor \times h \times w \times c_m$
$z^{l,(t_1)}$
Realign
$\lfloor \frac{T}{t_m} \rfloor \times h \times w \times c_m$
$z^{l,(t_m)}$
$h \times w \times c$
3D Conv
GN
Relu

Figure 4: Temporal-view Feature Fusing (TFF) block.

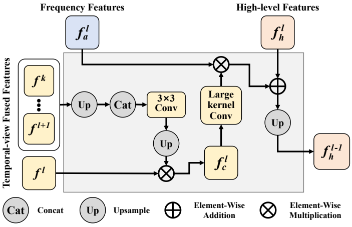

Figure 5: Multi-source Feature Fusing (MFF) block.

[29, 37], we propose a Multi-pyramid architecture to fully utilize different features.

Temporal-view Pyramid. The final features from the encoder likely represent high-level information of inpainting traces but lack detailed information, which impedes their detection ability when applied in complicated scenarios. Therefore, we integrate the intermediate features of different temporal views into the decoder to provide more instructive guidance.

Firstly, we develop a *Temporal-view Feature Fusing Block* (TFF) to fuse features from different views at the same stage. Denote $\mathbf{z}^{l,(t_1)},\dots,\mathbf{z}^{l,(t_m)}$ as the intermediate features of the temporal-view from t_1 to t_m at l-th stage, where $\mathbf{z}^{l,(t_1)} \in \mathbb{R}^{\lfloor \frac{T}{t_1} \rfloor \times h \times w \times c_1},\dots,\mathbf{z}^{l,(t_m)} \in \mathbb{R}^{\lfloor \frac{T}{t_m} \rfloor \times h \times w \times c_m}$ respectively. Denote v as the maximum of $\{\lfloor \frac{T}{t_1} \rfloor,\dots,\lfloor \frac{T}{t_m} \rfloor\}$. To fuse these features, this block first realigns their dimensions by expanding the first dimension to v. Taking $\mathbf{z}^{l,(t_1)}$ for instance, its size is aligned as $\lfloor \frac{T}{t_1} \rfloor \times h \times w \times c_1 \rightarrow v \times h \times w \times c_1$. Then we concatenate these features along the channel dimension and send them into a 3D convolution layer, which is followed by a Group Normalization to generate the fuse feature $\boldsymbol{f}^l \in \mathbb{R}^{h \times w \times c}$. For k stages in our method, we obtain a set of fused features $\{\boldsymbol{f}^1,\dots,\boldsymbol{f}^k\}$. These operations are shown in Fig. 4.

Frequency-assistance Pyramid. Inspired by the successful applications of frequency features in forensics [8, 26, 36], we extract the frequency features as an auxiliary to enhance detection. Given the video clip $\mathcal{V} = [\mathcal{I}_1,\dots,\mathcal{I}_T]$. We use the middle frame as a target to extract frequency features. We first employ Discrete Cosine Transform (DCT) [1] to convert this frame into frequency maps. Then we employ three frequency band-pass filters to decompose the frequency map into low-pass, mid-pass, and high-pass signals [26]. Based on these signals, we employ Inversed Discrete Cosine Transform (IDCT) to transform each signal as a frequency feature and concatenate all the transformed features along the channel dimension as the final frequency features, denoted as $\boldsymbol{f}_a^0 \in \mathbb{R}^{H \times W \times 3C}$. Then we build a frequency pyramid by adopting AvgPooling to reduce the size progressively as $\{\boldsymbol{f}_a^1,\dots,\boldsymbol{f}_a^k\}$.

Multi-source Feature Fusing Pyramid. We propose a *Multi-source Feature Fusing Block* (MFF) to fuse the temporal-view features, frequency features, and high-level features in a pyramid manner (see Fig. 5). For temporal-view features, at the l-th stage, we upsample and concatenate all deeper features and send them into a convolution layer. Then the dot-production is performed with the feature of the current stage. Performing dot-production instead of summation can instruct the current feature to focus on more important spots represented by the deep accumulated features. After these operations, we obtain accumulated features of k stages as $\{\boldsymbol{f}_c^1,\dots,\boldsymbol{f}_c^k\}$. We then fuse frequency features in an attention fashion. Note that $\boldsymbol{f}_a^l$ has the same height and width with $\boldsymbol{f}_c^l$. For each stage, we fuse the semantic features $\boldsymbol{f}_c^l$ with the frequency feature $\boldsymbol{f}_a^l$ as follows. We first perform a large kernel convolution [24] on $\boldsymbol{f}_c^l$ and then employ dot-production with $\boldsymbol{f}_a^l$. Denote the features after these operations as $\{\boldsymbol{f}_{c'}^{1},\ldots,\boldsymbol{f}_{c'}^{k}\}$. The motivation is that the frequency feature can instruct the semantic feature paying more attention to the frequency-importance spots.

Since the high-level features can provide important guidance, we fuse them with other features and convey them to the next pyramid layer. Denote the output features from the encoder as $\boldsymbol{f}_h$. They are integrated into the last obtained feature of k-th stage $\boldsymbol{f}_{c'}^{k}$ as $\boldsymbol{f}_h^{k}$ to provide high-level guidance. Then $\boldsymbol{f}_h^{k}$ is upsampled and sent into the next pyramid layer. The integration process is iterative, which builds another pyramid structure.

3.4 Objective Functions

Our objectives consist of two components. The first component measures the loss of mean Intersection-over-Union (mIoU) between the predicted detection mask $\mathcal{M}$ and the ground truth mask $\mathcal{M}_{\mathrm{GT}}$, which can be written as

$$\mathcal{L}_1(\mathcal{M},\mathcal{M}_{\mathrm{GT}}) = 1 - \frac{\sum(\mathcal{M}\cdot\mathcal{M}_{\mathrm{GT}})}{\sum(\mathcal{M}+\mathcal{M}_{\mathrm{GT}}-\mathcal{M}\cdot\mathcal{M}_{\mathrm{GT}})}. \tag{3}$$

The second component is a focal cross-entropy loss. Observing that the inpainted region is usually less than the authentic region. Thus we use focal loss here to mitigate this imbalance. This loss term can be defined as

$$\begin{aligned}\mathcal{L}_2(\mathcal{M},\mathcal{M}_{\mathrm{GT}}) = -\sum(&\alpha\cdot(1-\mathcal{M})^{\gamma}\cdot\mathcal{M}_{\mathrm{GT}}\log(\mathcal{M})\\ &+(1-\alpha)\mathcal{M}^{\gamma}(1-\mathcal{M}_{\mathrm{GT}})\log(1-\mathcal{M})),\end{aligned} \tag{4}$$

where α is the weight parameter to balance inpainted and authentic pixels and γ is the parameter for hard mining. The overall objectives can be expressed as $\mathcal{L}(\mathcal{M},\mathcal{M}_{\mathrm{GT}}) = \lambda_1\mathcal{L}_1(\mathcal{M},\mathcal{M}_{\mathrm{GT}}) + \lambda_2\mathcal{L}_2(\mathcal{M},\mathcal{M}_{\mathrm{GT}})$, where λ_1,λ_2 are the weight parameters to balance the losses.

4 Youtube-vos Video Inpainting Dataset

Davis Video Inpainting dataset (DVI) and the Free-from Video Inpainting dataset (FVI) are two existing datasets widely used in previous works [30, 36, 40]. DVI is constructed on DAVIS 2016 [25] using inpainting methods VI [12], OP [21] and CP [13] respectively. Each inpainting method corresponds to 50 videos. FVI dataset contains 100 test videos that are processed by object removal, and are usually used for demonstrating detection generalization [4]. In this paper, we introduce a more challenging and large-scale *Youtube-vos Video Inpainting dataset (YTVI)* for a more comprehensive assessment. This dataset is built upon Youtube-vos 2018 [33], which contains 3471 videos with 5945 object instances in its training set. Since only the training set of this dataset is fully annotated, we use it to construct YTVI. Specifically, with the goal of further improving the comprehensiveness, we adopt many more recent video inpainting methods on this dataset, including EG2 [17], FF [18], PP [41], and ISVI [38], together with VI, OP and CP. These inpainting methods are applied to the object regions annotated by ground truth masks.

5 Experiments

Datasets and Metrics. Our method is evaluated on the DVI and FVI datasets, as well as the newly proposed YTVI dataset. The detection performance is evaluated using the mean

Table 1: Performance of different methods on YTVI dataset. (a*,b*,c) denotes each method is trained on two inpainting methods (a,b), and tested on all inpainting methods (a,b,c).

Methods	**VI***	**OP***	**CP**	**VI**	**OP***	**CP***	**VI***	**OP**	**CP***
	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1
HPF (ICCV'19) [15]	0.50/0.63	0.48/0.59	0.42/0.54	0.12/0.18	0.47/0.59	0.52/0.64	0.52/0.65	0.13/0.20	0.55/0.69
GSRNet (AAAI'20) [39]	0.51/0.64	0.50/0.63	0.38/0.50	0.14/0.22	0.51/0.63	0.62/0.73	0.49/0.62	0.21/0.32	0.55/0.68
VIDNet (BMVC'21) [40]	0.62/0.74	0.51/0.64	0.43/0.56	0.15/0.23	0.54 /0.66	0.62/0.73	0.62/0.74	0.20/0.28	0.61/0.72
FAST (ICCV'21) [36]	0.49/0.61	0.54/0.66	0.46/0.58	0.25/0.35	0.47/0.58	0.60/0.71	0.47/0.59	0.29/0.40	0.62/0.73
OSNet (CVPR'22) [31]	0.60/0.70	0.58/0.67	0.56/0.66	0.17/0.23	0.61/0.71	0.69/0.78	0.65/0.74	0.30/0.40	0.70/0.78
HiFi-Net (CVPR'23)[9]	0.64/0.74	0.35/0.46	0.39/0.50	0.17/0.24	0.34/0.44	0.57/0.67	0.62/0.73	0.08/0.11	0.50/0.60
IML-ViT (AAAI'24) [20]	0.60/0.72	0.56/0.68	0.55/0.67	0.22/0.32	0.60/0.71	0.69/0.79	0.60/0.71	0.25/0.35	0.66/0.76
Ours	**0.72/0.82**	**0.67/0.78**	**0.63/0.75**	**0.33/0.45**	**0.70/0.79**	**0.75/0.84**	**0.73/0.82**	**0.42/0.54**	**0.73/0.83**

Table 2: Performance of different methods on DVI dataset.

Methods	**VI***	**OP***	**CP**	**VI**	**OP***	**CP***	**VI***	**OP**	**CP***
	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1	mIoU/F1
HPF (ICCV'19) [15]	0.46/0.57	0.49/0.62	0.46/0.58	0.34/0.44	0.41 /0.51	0.68/0.77	0.55/0.67	0.19/ 0.29	0.69/0.80
GSRNet (AAAI'20) [39]	0.57/0.69	0.50/0.63	0.51/0.63	0.30 /0.43	0.74/0.82	0.80/0.85	0.59 /0.70	0.22/0.33	0.70/0.77
VIDNet (BMVC'21) [40]	0.59/0.70	0.59/0.71	0.57/0.69	0.39/0.49	0.74/0.82	0.81/0.87	0.59/0.71	0.25/0.34	0.76/0.85
FAST (ICCV'21) [36]	0.61/0.73	0.65/0.78	0.63/0.76	0.32/0.49	0.78/0.87	0.82/0.90	0.57/ 0.68	0.22/0.34	0.76/0.83
DSTT (ICASSP'22) [30]	0.60/0.73	0.69/0.80	0.65/0.77	-	-	-	-	-	-
OSNet (CVPR'22) [31]	0.64/0.76	0.49/0.63	0.60/0.73	0.63/0.75	0.54/0.68	0.65/0.77	0.68/0.79	0.36/0.50	0.65/0.77
HiFi-Net (CVPR'23) [9]	0.71/0.81	0.80/0.88	**0.74/0.83**	0.65/0.76	0.80/0.88	0.83/0.90	0.72/0.82	0.42/0.54	**0.83/0.90**
IML-ViT (AAAI'24) [20]	**0.75/0.84**	0.69/0.80	0.71/0.82	0.68/0.80	0.69/0.80	0.75/0.85	**0.75/0.84**	0.52/0.66	0.75/0.84
Ours	0.71/0.81	**0.80/0.88**	0.70/0.82	**0.69/0.80**	**0.82/0.89**	**0.83/0.90**	0.72/0.82	**0.65/0.76**	0.81/0.89

Intersection-over-Union (mIoU) and F1 score as in previous works. For clarification, the threshold used in mIoU and F1 score is set to 0.5 for a fair comparison.

Implementation Details. Our method is implemented using PyTorch [23] with a GeForce RTX 3090Ti. For the backbone of the multilateral temporal-view encoder, we develop three branches and employ the Tiny, Small, and Base variants of the Swin Transformer in each branch. We use ViT Base architecture for the global encoder. The input size of video clips is 224×224 with the sequential length of 3 and is augmented by various common operations. In the training phase, we set the batch size to 12, and employ an SGD optimizer with a learning rate of 0.001 for the encoder and 0.01 for the decoder. We set the weight decay to 10^{-4} and use the poly learning rate decay to adjust the learning rate from the initialization to 10^{-5}.

Results on YTVI and DVI. Table 1 shows the performance of different detection methods on the YTVI dataset under both In-inpainting and Cross-inpainting evaluation. These methods are trained on two inpainting methods (marked *) and tested on all methods. For example, (VI*, OP*, CP) indicates that the method is trained on videos manipulated by VI and OP, and then tested on VI, OP, and CP respectively. It can be seen that our method outperforms others by a large margin. For example, in the setting of (VI*, OP*, CP), our method improves around 10.5% (mIoU) and 11.5% (F1) under averaged In-inpainting evaluation, and 7% (mIoU) and 9% (F1) under Cross-inpainting evaluation, compared to the second-best method OSNet. A similar trend is also observed in the other training settings (Right two groups in Table 1). Since the YTVI dataset is composed of real-world videos with high diversity, the proposed flexible collaboration ways of spatial-temporal clues exhibit significant advantages in comparison with the existing counterparts. Table 2 shows the performance of different methods on DVI dataset, revealing a similar trend as in Table 1.

Results on DVI → FVI. Following previous works [36, 40], we also evaluate the cross-dataset performance from DVI to FVI dataset. Table 3 shows the performance of each method trained on VI+OP in DVI dataset and tested on FVI dataset. It can be seen that our method outperforms others by a large margin, $\sim 6\%$ both in mIoU and F1 scores. Note that

Table 3: Cross-dataset performance of different methods from DVI to FVI dataset (DVI → FVI).

Methods	**DVI**	**FVI** mIoU/F1
HPF [15]	VI+OP	0.20/0.28
GSR-Net [59]		0.19/0.28
VIDNet [40]		0.25/0.36
FAST [56]		0.28/0.35
OSNet [51]		0.26/0.38
HiFi-Net [9]		0.13/0.19
IML-ViT [20]		0.29/0.42
Ours		**0.36/0.48**

Table 4: Cross-dataset performance of different methods from YTVI to DVI dataset (YTVI → DVI).

Methods	**YTVI**	**DVI** **VI** mIoU/F1	 **OP** mIoU/F1	 **CP** mIoU/F1
HPF [15]	VI+OP	0.22/0.32	0.37/0.49	0.40/0.52
GSRNet [59]		0.17/0.26	0.43/0.56	0.43/0.56
VIDNet [40]		0.17/0.27	0.29/0.41	0.29/0.42
FAST [56]		0.38/0.50	0.59/0.71	0.53/0.67
OSNet [51]		0.35/0.44	0.49/0.61	0.50/0.62
HiFi-Net [9]		0.0003/0.0006	0.65/0.76	0.39/0.49
IML-ViT [20]		0.37/0.48	0.53/0.66	0.41/0.52
Ours		**0.44/0.55**	**0.69/0.80**	**0.64/0.76**

Table 5: Cross-dataset Cross-inpainting Performance of different methods from YTVI to DVI (YTVI → DVI).

Methods	**YTVI** **FF*** mIoU/F1	 **EG2*** mIoU/F1	 **PP*** mIoU/F1	 **IS** mIoU/F1	 **VI** mIoU/F1	 **OP** mIoU/F1	 **CP** mIoU/F1	**DVI** **VI** mIoU/F1	 **OP** mIoU/F1	 **CP** mIoU/F1
HPF [15]	0.47/0.58	0.41/0.52	0.39/0.50	0.22/0.33	0.14/0.22	0.11/0.18	0.20/0.30	0.28/0.40	0.12/0.19	0.39/0.52
GSRNet [59]	0.71/0.81	0.65/0.77	0.60/0.72	0.30/0.44	0.06/0.11	0.15/0.24	0.19/0.29	0.61/0.73	0.31/0.44	0.66/0.78
VIDNet [40]	0.56/0.68	0.50/0.63	0.47/0.59	0.23/0.34	0.13/0.20	0.16/0.24	0.31/0.43	0.37/0.50	0.20/0.28	0.37/0.49
FAST [56]	0.54/0.66	0.52/0.63	0.48/0.60	0.30/0.43	0.19/0.29	0.26/0.36	0.37/0.49	0.52/0.65	0.40/0.52	0.55/0.68
OSNet [51]	0.74/0.82	0.64/0.74	0.69/0.78	0.42/0.54	0.20/0.29	0.30/0.39	0.43/0.54	0.65/0.77	0.49/0.62	0.65/0.77
HiFi-Net [9]	0.50/0.62	0.43/0.56	0.33/0.45	0.46/0.59	0.15/0.23	0.08/0.13	0.15/0.22	0.61/0.73	0.49/0.61	0.71/0.81
IML-ViT [20]	0.71/0.81	0.67/0.78	0.64/0.75	0.40/0.54	0.19/0.28	0.35/0.46	**0.54/0.67**	0.59/0.72	0.50/0.63	0.64/0.76
Ours	**0.77/0.86**	**0.73/0.83**	**0.69/0.80**	**0.51/0.65**	**0.27/0.38**	**0.36/0.48**	0.47/0.60	**0.67/0.79**	**0.67/0.79**	**0.73/0.84**

Table 6: Effect of each component.

Base	**TF**	**FF**	**MT**	**DWTI**	**VI*** mIoU/F1	**OP** mIoU/F1	**CP*** mIoU/F1
✓					0.64/0.76	0.55/0.68	0.69/0.80
✓	✓				0.71/0.81	0.59/0.72	0.79/0.87
✓	✓	✓			0.70/0.81	0.60/0.73	0.80/0.88
✓	✓	✓	✓		0.72/0.82	0.63/0.75	0.82/0.89
✓	✓	✓	✓	✓	0.72/0.82	0.65/0.76	0.81/0.89

Table 7: Temporal-view selection analysis.

Selection	**VI*** mIoU/F1	**OP** mIoU/F1	**CP*** mIoU/F1
View 1	0.71/0.82	0.61/0.63	0.80/0.88
View 2	0.71/0.81	0.59/0.72	0.80/0.88
View 3	0.71/0.81	0.57/0.69	0.81/0.88
View 1,2	0.72/0.82	0.62/0.74	0.82/0.89
View 1,3	0.72/0.82	0.64/0.76	0.82/0.89
View 2,3	0.71/0.81	0.59/0.71	0.81/0.89
View 1,2,3	0.72/0.82	0.65/0.76	0.81/0.89

Table 8: Complexity Analysis.

Method	**OP** mIoU/F1	**FLOPs** (G)	**Params** (M)	**Throughput** (imgs/s)
VIDNet[40]	0.20/0.31	164.7	82.8	36.9
FAST [56]	0.22/0.34	96.0	312.3	100.5
HiFi-Net [9]	0.42/0.54	62.9	10.1	81.2
IML-ViT [20]	0.52/0.66	445.6	88.6	8.8
Mumpy-♠	0.62/0.74	48.0	262.6	91.2
Mumpy-♣	0.64/0.75	67.0	318.7	61.2
Mumpy-♠♣	0.59/0.71	29.9	237.9	98.1
Mumpy	0.65/0.76	89.2	361.3	59.1

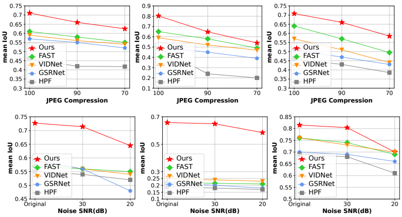

Figure 6: Robustness evaluation.

FAST and VIDNet are video-based methods, which show better generalization performance than others. It confirms the significance of modeling the temporal dependencies.

Results on YTVI → DVI. Table 4 shows another cross-dataset experiment from YTVI to DVI dataset. Similarly, we use VI + OP in the YTVI dataset for training and evaluate all inpainting methods on the DVI dataset. We can observe our method outperforms all counterparts, improving 10% on average in mIoU and F1 scores. Moreover, we perform a more challenging evaluation under cross-dataset cross-inpainting in Table 5. Specifically, we use inpainting methods FF, EG2, and PP in the YTVI dataset for training and evaluate all inpainting methods on the YTVI and DVI datasets. The results show our method outperforms all the counterparts by a large margin, averaging 6.4% and 6.2% in mIoU and F1 scores compared with the second-best IML-ViT.

Ablation Study. This study is performed on on the DVI dataset. **1) Effect of each component.** Denote Base as only using one branch as the encoder with a traditional upsampling decoder. TF denotes using the intermediate temporal-view features in the decoder. FF denotes using the frequency features in the decoder. MT denotes using the multilateral temporal-view settings. DWTI denotes using the deformable window-based temporal-view interaction. The results in Table 6 reveal the performance is gradually improved under both the in-domain and cross-domain scenarios by adding each component. **2) Temporal-view Selection Analysis.** We study the effect of using only one view and two views in Table 7. The results show that using all views can notably improve the cross-inpainting detection performance on OP, *i.e.*, achieving better generalization ability. **3) Complexity Analysis.** Table 8 shows the computational cost and throughput of different methods. Mumpy-♠ denotes using Swin-Tiny for all views in encoder. Mumpy-♣ means using a compact decoder by downsampling all features to their 1/2. Mumpy-♠♣ denotes using both Swin-Tiny and compact decoder. It can be seen that despite having more parameters, our method has less FLOPs than the compared methods while showing much better cross-inpainting performance. By reducing the model size, our method still outperforms others by a large margin with much less FLOPs.

Robustness Analysis. Following [36], we train the methods on VI+OP and test them on VI, OP, and CP in the JPEG compression using the quality factors of 70 and 90. Moreover, we train the methods on VI+CP and test them on VI, OP, and CP in the Gaussian noise with the signal-to-noise ratios (SNR) from 20 to 30 dB. The results are depicted in Fig. 6, showing that despite all methods being degraded with increased perturbations, our method drops slower, showing the robustness of our method to resist common distortions.

Sanity Check. We assess the ability of our method to distinguish between authentic and inpainted frames. We perform experiments on the DVI dataset and train methods using VI+OP, and average the pixel-wise predictions as the frame-level result. As shown in Table 9. We can observe that our method performs best compared with others on all inpainting methods, demonstrating that our method can learn the discriminative inpainting clues. Surprisingly, HiFi-Net does not perform as expected. It may be because of the inappropriate margin setting between authentic and inpainted pixels, leading to a greater concentration on authentic pixels.

Table 9: Sanity check for image-level classification AUC comparison on DVI dataset.

Methods	**VI***	**OP***	**CP**
HPF	0.718	0.640	0.845
GSRNet	0.762	0.758	0.834
VIDNet	0.778	0.768	0.884
FAST	0.795	0.787	0.898
OSNet	0.992	0.981	0.989
HiFi-Net	0.642	0.699	0.682
Ours	0.996	0.993	0.997

6 Conclusion

This paper introduces a new Transformer (*MumPy*) to flexibly collaborate spatial-temporal clues. To achieve this, we develop a multilateral temporal-view encoder to extract various collaborations of spatial-temporal clues and propose a deformable window-based temporal-view interaction module to enhance diversity. We then describe a multi-pyramid decoder to generate detection maps by aggregating various types of features. Our method is validated on existing datasets and a new proposed challenging Youtube-vos video inpainting dataset. The results demonstrate the efficacy of our method in both in-domain and cross-domain evaluation scenarios.

Acknowledgement: This work was supported in part by the National Natural Science Foundation of China under Grant No.62402464, China Postdoctoral Science Foundation under Grant No.2021TQ0314 and Grant No.2021M703036.

References

[1] Nasir Ahmed, T_ Natarajan, and Kamisetty R Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.

[2] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021.

[3] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, volume 2, page 4, 2021.

[4] Ya-Liang Chang, Zhe Yu Liu, Kuan-Ying Lee, and Winston Hsu. Free-form video inpainting with 3d gated convolution and temporal patchgan. *In Proceedings of the International Conference on Computer Vision (ICCV)*, 2019.

[5] Chengbo Dong, Xinru Chen, Ruohan Hu, Juan Cao, and Xirong Li. Mvss-net: Multi-view multi-scale supervised networks for image manipulation detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3539–3553, 2022.

[6] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6824–6835, 2021.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[8] Qiqi Gu, Shen Chen, Taiping Yao, Yang Chen, Shouhong Ding, and Ran Yi. Exploiting fine-grained face forgery clues via progressive enhancement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 735–743, 2022.

[9] Xiao Guo, Xiaohong Liu, Zhiyuan Ren, Steven Grosz, Iacopo Masi, and Xiaoming Liu. Hierarchical fine-grained image forgery detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3155–3165, 2023.

[10] GM Harshvardhan, Mahendra Kumar Gourisaria, Manjusha Pandey, and Siddharth Swarup Rautaray. A comprehensive survey and analysis of generative models in machine learning. *Computer Science Review*, 38:100285, 2020.

[11] Shijia Huang, Yilun Chen, Jiaya Jia, and Liwei Wang. Multi-view transformer for 3d visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15524–15533, 2022.

[12] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Deep video inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5792–5801, 2019.

[13] Sungho Lee, Seoung Wug Oh, DaeYeun Won, and Seon Joo Kim. Copy-and-paste networks for deep video inpainting. In *International Conference on Computer Vision (ICCV)*, 2019.

[14] Ang Li, Qiuhong Ke, Xingjun Ma, Haiqin Weng, Zhiyuan Zong, Feng Xue, and Rui Zhang. Noise doesn't lie: towards universal detection of deep inpainting. *arXiv preprint arXiv:2106.01532*, 2021.

[15] Haodong Li and Jiwu Huang. Localization of deep inpainting using high-pass fully convolutional network. In *proceedings of the IEEE/CVF international conference on computer vision*, pages 8301–8310, 2019.

[16] Yuanman Li, Liangpei Hu, Li Dong, Haiwei Wu, Jinyu Tian, Jiantao Zhou, and Xia Li. Transformer-based image inpainting detection via label decoupling and constrained adversarial training. *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[17] Zhen Li, Cheng-Ze Lu, Jianhua Qin, Chun-Le Guo, and Ming-Ming Cheng. Towards an end-to-end framework for flow-guided video inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[18] Rui Liu, Hanming Deng, Yangyi Huang, Xiaoyu Shi, Lewei Lu, Wenxiu Sun, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fuseformer: Fusing fine-grained information in transformers for video inpainting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14040–14049, 2021.

[19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[20] Xiaochen Ma, Bo Du, Xianggen Liu, Ahmed Y Al Hammadi, and Jizhe Zhou. Iml-vit: Image manipulation localization by vision transformer. *arXiv preprint arXiv:2307.14863*, 2023.

[21] Seoung Wug Oh, Sungho Lee, Joon-Young Lee, and Seon Joo Kim. Onion-peel networks for deep video completion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4403–4412, 2019.

[22] Jungin Park, Jiyoung Lee, and Kwanghoon Sohn. Dual-path adaptation from image to video transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2203–2213, 2023.

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[24] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4353–4361, 2017.

[25] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016.

[26] Yuyang Qian, Guojun Yin, Lu Sheng, Zixuan Chen, and Jing Shao. Thinking in frequency: Face forgery detection by mining frequency-aware clues. In *European conference on computer vision*, pages 86–103. Springer, 2020.

[27] Jingjing Ren, Qingqing Zheng, Yuanyuan Zhao, Xuemiao Xu, and Chen Li. Dlformer: Discrete latent transformer for video inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3511–3520, 2022.

[28] Zhihao Sun, Haoran Jiang, Danding Wang, Xirong Li, and Juan Cao. Safl-net: Semantic-agnostic feature learning network with auxiliary plugins for image manipulation detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22424–22433, 2023.

[29] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 568–578, 2021.

[30] Shujin Wei, Haodong Li, and Jiwu Huang. Deep video inpainting localization using spatial and temporal traces. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8957–8961. IEEE, 2022.

[31] Haiwei Wu, Jiantao Zhou, Jinyu Tian, and Jun Liu. Robust image forgery detection over online social network shared images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13440–13449, 2022.

[32] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision transformer with deformable attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4794–4803, 2022.

[33] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 585–601, 2018.

[34] Shen Yan, Xuehan Xiong, Anurag Arnab, Zhichao Lu, Mi Zhang, Chen Sun, and Cordelia Schmid. Multiview transformers for video recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3333–3343, 2022.

[35] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 2022.

[36] Bingyao Yu, Wanhua Li, Xiu Li, Jiwen Lu, and Jie Zhou. Frequency-aware spatiotemporal transformers for video inpainting detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8188–8197, 2021.

[37] Dong Zhang, Hanwang Zhang, Jinhui Tang, Meng Wang, Xiansheng Hua, and Qianru Sun. Feature pyramid transformer. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pages 323–339. Springer, 2020.

[38] Kaidong Zhang, Jingjing Fu, and Dong Liu. Inertia-guided flow completion and style fusion for video inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5982–5991, June 2022.

[39] Peng Zhou, Bor-Chun Chen, Xintong Han, Mahyar Najibi, Abhinav Shrivastava, Ser-Nam Lim, and Larry Davis. Generate, segment, and refine: Towards generic manipulation segmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13058–13065, 2020.

[40] Peng Zhou, Ning Yu, Zuxuan Wu, Larry S Davis, Abhinav Shrivastava, and Ser Nam Lim. Deep video inpainting detection. In *BMVC*, 2021.

[41] Shangchen Zhou, Chongyi Li, Kelvin C.K Chan, and Chen Change Loy. ProPainter: Improving propagation and transformer for video inpainting. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2023.