基于卷积 LSTM 的残差网络用于深度伪造视频检测

Shahroz Tariq、Sangyup Lee
成均馆大学计算机科学与工程系
韩国水原
{shahroz,sangyup.lee}@g.skku.edu

Simon S. Woo*
韩国水原成均馆大学应用数据科学系
swoo@g.skku.edu

摘要

近年来，基于深度学习的视频操纵方法已被大众广泛接受。几乎不费吹灰之力，人们就能轻松学会如何只用几个受害者或目标图像生成深度伪造视频。这给照片在互联网（尤其是社交媒体网站）上公开的每个人带来了严重的社会问题。目前已开发出几种基于深度学习的检测方法来识别这些深度伪造视频。然而，这些方法缺乏通用性，因为它们只对特定类型的深度伪造方法表现良好。因此，这些方法无法用于检测其他深度伪造方法。此外，它们也没有利用视频的时间信息。在本文中，我们解决了这些局限性。我们开发了一种基于卷积 LSTM 的残差网络 (CLRNet)，它将连续图像序列作为视频的输入，学习时间信息，帮助去除深度伪造视频帧与帧之间存在的不自然外观伪影。我们还提出了一种基于迁移学习的方法来推广不同的深度伪造方法。通过使用 FaceForensics++ 数据集进行的严格实验，我们发现我们的方法优于之前提出的五种最先进的深度防伪检测方法，因为它能更好地泛化使用相同模型检测不同深度防伪方法的能力。

关键词

深度伪造检测、视频取证、图像处理

1 引言

基于深度学习的合成图像生成方法在过去几年中蓬勃发展。这些新方法能生成逼真的图像，可以轻松欺骗普通人[15, 24, 25, 35, 40, 41]。由于它们的能力，这些方法在计算机视觉或图形学科中有很多应用，如人脸生成 [24] 和逼真景物生成 [31]。然而，所有这些创新也有其阴暗面。许多怀有恶意的人利用这些方法生成假的名人和大众视频 [11, 13, 14, 23]、
有许多方法可以解决这一问题 [15、25、40、41]。这已经开始引发重大的社会问题：最近的一项研究称，96% 的深度伪造源于色情视频[30]。它们都属于所谓的 "深度伪造"（Deepfakes）。最近，重新搜索社区发布了许多深度假冒数据集，以帮助其他研究人员开发这些深度假冒的检测机制。其中最具开创性的工作是 FaceForensics++ 数据集[35]，部分由谷歌开发。最初，FaceForen-sics++[35] 数据集包含 Pristine（1,000 个）、Deepfakes（1,000 个）、Deepfakes（1,000 个）、Deepfakes（1,000 个）、
FaceSwap (1,000), Face2Face (1,000) 和 Neural Texture (1,000) 视频。后来，谷歌也加入了真视频（363 个）和假视频（3000 个）。今年，Facebook 发起了一项奖金为一百万美元的深度识假挑战，以加速该领域的研究[6]。最近，出现了几种在特定训练 deepfake 数据集上具有较高零点测试准确率的 deepfake 检测方法 [3, 10, 21, 22, 29, 39]。但是，这些方法对训练集中不存在的新的 deepfake 方法的检测准确率较低。这就是我们开发通用和普遍的 deepfake 视频检测器的主要动机，因为为每一种新的 deepfake 生成方法制作数据集是不切实际的。因此，我们利用已有的大量深度伪造数据集，如 FaceForensics++ [35]，并采用迁移学习来检测其他深度伪造方法和新生成的方法。

关于开发通用深层伪造检测器的研究包括
的研究活动有限[10]。因此，我们的目标是通过开发一种模型来解决这些问题，该模型首先从 FaceForen- sics++ [35]的海量深度仿真数据集中训练一种深度仿真方法，然后使用少点转移学习法来学习其他深度仿真方法。少量学习所需的样本视频数量极少，因此选择这种方法更为实用。我们的方法与之前的研究不同，我们探索了不同的迁移学习策略，并通过在多个数据集上的严格实验比较了它们的结果。

我们注意到，大多数深度伪造检测方法 [10, 35] 都是从视频中随机提取帧（图像）进行训练和测试，因此是一种基于单帧的检测方法。然而，在仔细观察了大量 deepfake 视频之后，我们惊讶地发现，在 deepfake 视频中，连续帧之间存在微小的伪影，我们可以通过这些伪影来识别这些视频，如图 1 所示。因此，我们得出结论，连续视频帧之间的时间信息对于深度伪造检测至关重要[36]。为了结合时间方面的信息，我们使用了卷积 LSTM，因为它已被证明可用于此类任务[43]。因此，我们提出了 CLRNet，这是一种基于卷积 LSTM 的残差网络，可利用跨铁器学习技术进行深度假视频检测。

主要贡献概述如下

- **CLRNet**：我们提出了一种基于会话式 LSTM 和残差网络的新架构，用于深度伪造检测
  使用视频中的连续帧序列。
- **通用性**：与之前最先进的深度伪造检测方法相比，我们提供了一种更

*通讯作者

具通用性的方法

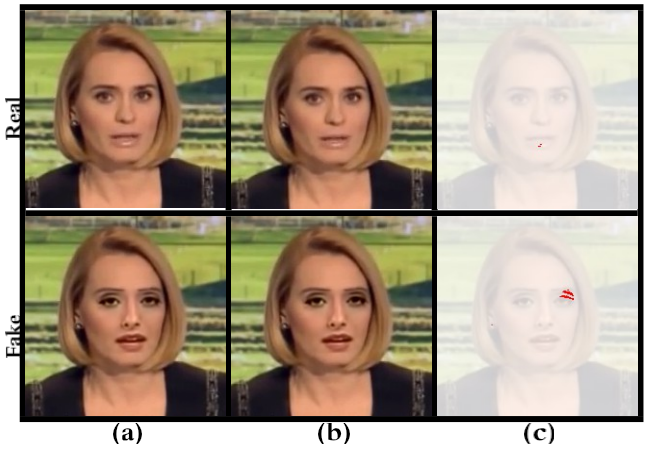


图 1：深度伪造视频的两个连续帧之间的差异。(a) 和 (b) 分别是第 n 和 $(n+1)$ 个帧。对于原始视频，这些帧几乎是相同，但对于深度伪造视频来说，存在不一致的地方。第 n 帧和第 $(n+1)$ 帧之间的差异突出显示了 (c) 中的不一致之处（红色）。差异很小但如 (c) 所示，假视频帧之间的差异更为明显。

并通过严格的实验证明了这一点。

2 相关工作

我们的工作横跨不同领域，如深度假货检测、模型泛化和迁移学习。因此，我们将在本节简要介绍相关工作。

2.1 深度伪造检测

对异常眨眼的检测[26]已被证明可有效识别被处理视频或图像中的不一致之处。此外，图像拼接检测方法[4, 20, 37]旨在利用图像中被操纵区域边界附近的拼接所产生的偏差。虽然现有的深度伪造生成方法生成的图像中的不一致性可以被检测出来，但每年都有新的更先进的生成方法被重新搜索和开发出来。另一方面，基于模型的方法，如测量去马赛克伪影[16]、镜头像差[45]和选择不同图像处理方法[2]所产生的 JPEG 伪影等特征，传统上一直被用于识别图像处理。然而，在检测机器生成的伪造图像（如生成对抗网络（GAN）和变异自动编码器（VAE））时，这些方法并不可靠，因为整套图像都是从零开始创建的。在有监督的环境中，基于深度学习的方法已显示出很高的检测精度。具体来说，基于卷积神经网络（CNN）的方法主要是从输入的 RGB 彩色图像中自动学习分层表示[27, 33]，或利用操纵检测特征[5]，并使用手工创建的图像[6]。

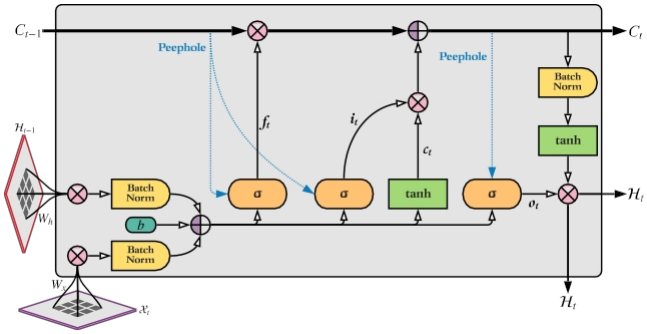


图 2：*卷积 LSTM 单元*：卷积 LSTM 单元的可视化表示。其主要结构与 LSTM 类似，但增加了一些组件以纳入卷积部分。这里，X_t 是输入，C_t 是单元输出，H_t 是隐藏状态。门由 i_t、f_t 和 o_t 表示。

特征[9]。Tariq 等人[38, 39]引入了 ShallowNet，这是一种基于 CNN 的快速学习和高效网络，即使在低分辨率（64×64）下也能高精度检测 GAN 生成的图像。毛皮
此外，Zhou 等人[47] 应用了一种双流快速 R-CNN 网络。
的工作，它可以捕捉高低层次的图像细节。Rössler 等人[34, 35]的研究显著提高了压缩图像的性能，这对于检测 Instagram、Facebook 和 Twitter 等社交网站上的深度伪造图像至关重要。上述方法大多集中于检测单帧视频中的面部操作。然而，如图 1 所示，分析深度伪造视频中连续帧之间的时间信息至关重要。在我们的方法中，我们使用多个连续帧来利用这些时间信息，从而改进对深度伪造视频的检测。

2.2 使用连续视频帧进行检测

Sabir 等人[36] 提出了一种检测方法，利用 CNN 和递归神经网络 (RNN) 捕获 5 个连续 deepfake 视频帧中呈现的时间信息。此外，Güera 等人[18]也采用了类似的方法，利用 CNN 层从多达 80 个连续帧中提取特征，并将其输入 RNN 层，从而建立了一个时间信息感知的深度伪造检测模型。这两种方法 [18, 36] 都是从 CNN 提取特征并将其传递给 RNN 层。同时，我们使用卷积 LSTM 单元建立 CLRNet 模型，它可以直接从输入图像序列中捕捉时空信息。然而，当在包含不同深度伪造生成方法视频的数据集上进行评估时，大多数方法的结果都较差。因此，我们对 CLRNet 模型进行了迁移学习，以应对这一挑战。

2.3 通过迁移学习进行推广

各种深度伪造视频生成技术不断被开发出来，未来还会出现更复杂的深度伪造视频。然而，收集和制作大量新的深度伪造样本是不切实际的。为了应对这种情况

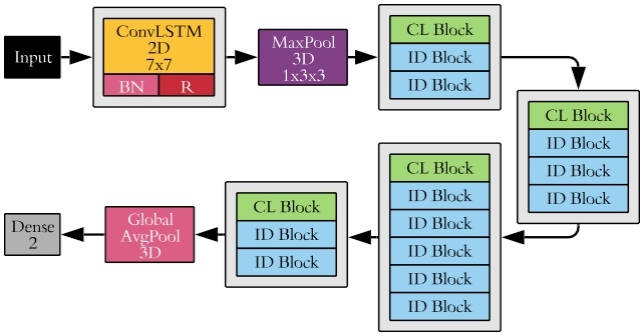


图 3：*CLRNet 架构*：基于卷积 LSTM 的残差网络 (CLRNet) 模型的高层架构图。模型的输入是连续图像序列，输出是真假分类结果。我们使用 Keras temrs 来表示层名。

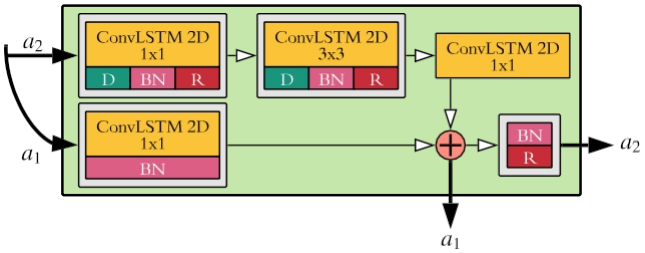


图 4：*CL 块*：卷积 LSTM 模块（CL 模块）内部结构的可视化表示。共有两个 ConvLSTM2D 层，每个层之后都有 dropout、BatchNorm 和 ReLU。之后，我们直接将最后一个 ConvLSTM2D 层添加到带有 BatchNorm 层的 ConvLSTM2D 中，通过快捷连接得到 $_1$。最后，BatchNorm 和 ReLU 应用于加法运算器的输出，*得到*。$_{21}$ 和 $_2$ 的输入对于第一个 CL 块是相同的，但之后则不同。

在这种情况下，少量转移学习（TL）是识别不同方法生成的深度伪造图像的关键。也就是说

在一个领域（如 FaceSwap）学到的知识可用于提高在另一个领域（如 Face2Face）的通用性。科佐利诺

等人 [10] 对单一检测方法在多个目标领域的通用性进行了实验。在这项工作中，我们比较了

我们的方法与 ForensicsTransfer [10] 进行了对比，以证明我们的方法具有更强的通用性和可移植性。

3 我们的方法

对深度伪造视频的逐帧分析揭示了深度伪造视频中连续帧之间的不一致性，而这些不一致性在原始视频中是不存在的。这些不一致之处包括：1）面部一小块区域的亮度和对比度突然变化；2）某些面部部位（如眼睛、嘴唇和眉毛）的大小在帧与帧之间发生变化。这些都是细微的不一致，只要仔细检查就能发现。图 1 显示了深层伪造视频中两个连续帧中的此类伪影示例，并用红色标出了它们之间的突然差异。这些不一致使视频变得有些不自然。受这一发现和观察结果的启发，我们开发了一种新的深度防伪方法--卷积 LSTM 残差网络（Convolutional LSTM Residual Network），它可以考虑这些不一致性，从而识别真假视频。

3.1 卷积 LSTM 单元

Shi 等人[43]指出，FC-LSTM[17]处理时空数据的主要问题是在输入到状态和状态到状态的转换过程中使用全连接，而不对空间形成进行编码。相比之下，卷积 LSTM（ConvL- STM）通过引入三维张量克服了这一问题。

分别用"◦"和"∗"表示。

$$i_t = \sigma\, W_x \mathsf{i} * \mathsf{X}_t + W_h \mathsf{i} * \mathsf{H}_{t-1} + W_c \mathsf{i} \circ C_{t-1} + b_i$$

$$f_t = \sigma\, W_x \mathsf{f} * \mathsf{X}_t + W_h \mathsf{f} * \mathsf{H}_{t-1} + W_c \mathsf{f} \circ C_{t-1} + b_f$$
$$C_t = f_t \circ C_{t-1} + i_t \circ \tanh W_x \mathsf{c} * \mathsf{X}_t + W_h \mathsf{c} * \mathsf{H}_{t-1} + \quad (1)$$
$$b_c$$

$$o_t = \sigma\, W_x \mathsf{o} * \mathsf{X}_t + W_h \mathsf{o} * \mathsf{H}_{t-1} + W_c \mathsf{o} \circ C_t + b_o$$

(X1, ... , Xt), 输出 (C1, ... , Ct), 隐藏状态 (H1, ... , Ht), 和门 ($_{it}$, $_{ft}$, $_{ot}$).在本文中，我们沿用了 Shi 等人 [43] 提出的 ConvL- STM。哈达玛乘积和卷积

$$H_t = o_t \circ \tanh(C_t)$$

此外，图 2 显示了我们基于 Xavier [42] 和 Keras [8] 实现的 ConvLSTM 单元的可视化表示。

3.2 卷积 LSTM 残差网络

Sabir 等人[36]的研究表明，基于 CNN 的骨干编码网络（如 ResNet 或 DenseNet）与基于 RNN 的网络连接，可实现高精度的深度伪造检测任务。他们还发现，与使用单帧输入相比，图像序列的性能更高。此外，为了避免梯度消失问题，我们在网络中加入了残差。基于之前的研究[28, 29, 36, 44]和我们对连续帧中不一致和/或伪影的分析，我们开发了基于卷积 LSTM 的残差网络（CLRNet）。图 3 显示了我们的 CLRNet 模型的可视化架构。我们模型的输入元素是三维张量，保留了连续帧的全部空间信息。因此，我们使用 ConvLSTM（CL）单元来代替卷积单元，如图 4 和图 5 所示。Shi 等人[43]指出，以这种方式堆叠 ConvLSTM 可为模型提供强大的表示能力。我们使用两种构建模块（即 CL 模块和 ID 模块）开发了 CLRNet 的核心架构。图 4 和图 5 是 CL 和 ID 模块的图示。这些模块分别与 ResNet [19] 中的卷积模块和标识模块相关。如图 4 和图 5 所示，我们的 CLRNet 模型中的构建模块有两个输出（即$_1$ 和 ）$_2$。在这两个模块中，$_1$ 是加法步骤后的输出，而$_2$ 是加法步骤后的输出，然后是批量归一化和 ReLU 层。$_1$ 和$_2$ 的输出是下一个区块的输入。CL 模块包含一个 ConvLSTM 单元，然后是一个批次归一化和 ReLU 层。

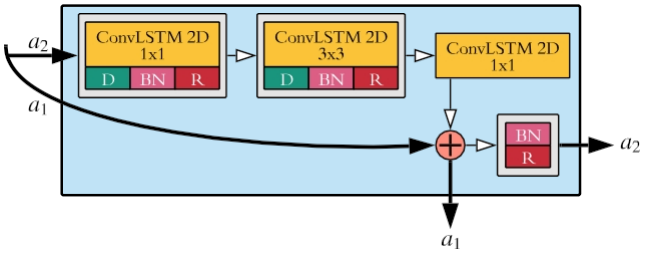


图 5：*ID 块*：身份区块（ID Block）内部结构的可视化表示。与 CL 数据块类似，ID 数据块也有两个 ConvLSTM2D 层，每个层之后都有 dropout、BatchNorm 和 ReLU。之后，我们将最后一个 ConvLSTM2D 层添加到来自快捷连接的输入 a_1，得到输出 a_1。最后，a_2 是 BatchNorm 和 ReLU 应用于加法运算器输出的结果。

而在 ID 模块中，快捷路径直接将输入$_1$ 连接到加法层。

3.3 迁移学习策略

在迁移学习方面，我们评估了以下三种策略：

1）单源到单目标：我们用一个大型深度伪造数据集训练模型，然后将迁移学习应用于一个目标深度伪造数据集（例如，首先在 DF 数据集上训练模型，然后将迁移学习应用于 FS）；2）多源到单目标：我们在多个源上训练模型，然后将迁移学习应用于单个目标域；3）单源到多目标：我们在单个源域上训练模型，然后使用少量多个目标域来应用迁移学习。我们将在第 4 节讨论每种策略的优缺点。

3.4 实施细节

*3.4.1 数据集描述。*为了将我们的方法与不同的基线进行比较，我们使用了 DeepFake (DF)、FaceSwap (FS)、Face2Face (FS)、NeuralTextures(NT) 和 DeepFakeDetection (DFD) 数据集。表 1 介绍了本文使用的所有数据集。除 DFD 外，FaceForensics++[35] 数据集中的每个类别都包含 1,000 个视频。我们将 1,000 个视频中的前 750 个用于训练，接下来的 125 个用于验证，剩下的 125 个用于测试。对于 DFD，我们只选择了 300 个视频（250 个用于训练，25 个用于验证，25 个用于测试）。

*3.4.2 预处理。*我们从每段真实和虚假视频中提取 16 个样本，每个样本包含五个连续帧。我们使用多任务 CNN（MTCNN）[46] 来检测提取帧内的人脸地标信息。然后，我们利用这些地标信息对图像中的人脸进行裁剪，并对齐人脸

。

使其居中。最后，我们将所有帧的大小调整为 240 × 240
决议。

*3.4.3 数据扩充。*我们还采用了数据扩增技术来丰富训练数据。我们改变了以下条件：1) 亮度（-30% 到 30%），2) 通道偏移（-50 到 50）、

表 1：*数据集详情*Pristine、 DeepFake、 FaceSwap、 Face2Face 和 Neural Textures 数据集分别有 1,000 个视频：我们使用 750 个真实视频和 750 个虚假视频进行训练，使用 125 个真实视频和 125 个虚假视频进行验证和测试。Deep FakeDetection 数据集中有 3,363 个视频（363 个真实视频，3,000 个虚假视频）：我们使用 250 个真实视频和 250 个虚假视频进行训练，使用 25 个真实视频和 25 个虚假视频进行验证和测试。在迁移学习中，我们使用了 10 个真实视频和 10 个虚假视频。

数据集	视频总数	基地培训视频	迁移学习视频	每个样本视频
质朴（真实）	1,000	750	10	16
DeepFake (DF)	1,000	750	10	16
FaceSwap (FS)	1,000	750	10	16
面对面 (F2F)	1,000	750	10	16
神经纹理 (NT)	1,000	750	10	16
深度防伪检测（DFD）	3,363	250	10	16

3) 缩放（-20% 到 20%），4) 旋转（-30°度到 30°），5) 水平翻转（50% 概率）。

*3.4.4 训练*我们的方法背后的理念是，先在一个广泛可用的深度假数据集上进行训练，然后使用迁移学习，以少量样本对其他数据集进行训练。我们从 750 个真实视频（Pristine）和 750 个伪造视频（DF 或 FS 或 F2F 或 NT）中各抽取 16 个样本来训练模型。

*3.4.5 迁移学习配置。*基础模型的训练完成后，我们使用目标数据集的一小部分子集（10 个视频）对其他数据集进行迁移学习。对于我们的 CLRNet 模型，我们冻结了前 120 层，并使用来自源数据集和目标数据集的相同数量视频（即每个数据集 10 个视频）对模型进行迁移学习。

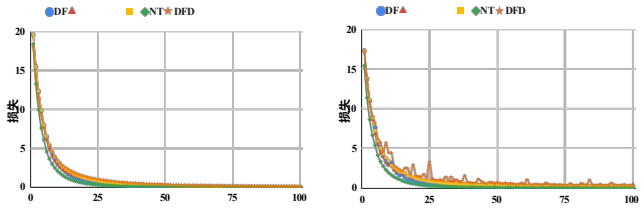
*3.4.6 机器配置。*我们使用 Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz，256.0GB 内存和 NVIDIA GeForce Titan RTX。我们使用 TensorFlow v1.13.0 [1] 和 Keras Library [8] on Python v3.7.5 来实现 CLRNet 模型。

*3.4./ 评估指标。*我们使用精度、召回率和 F1 分数进行评估。由于篇幅有限，我们在表 2 和表 3 中只报告了 F1 分数。ShallowNet、Xception、FF++ [35] 和 FT 使用单帧作为训练集和测试集的输入，而 Sabir *等人* [36] 和我们的 CLRNet 使用五个连续帧作为输入。我们在训练集、验证集和测试集中保持了相同数量的真实和虚假图像，以尽量减少评估过程中数据不平衡的影响。

3.5 基准方法

我们将 CLRNet 与几种最先进的方法进行了比较，并尽力按照它们的规格来实现。以下是对这些方法的描述。

*3.5.1 Xception*Xception 网络[7]被认为是图像分类任务中最先进的网络。我们使用了 Xception 的 Keras [8] 实现，它是在 ImageNet 数据集 [12] 上预先训练好的。



(a) 培训损失 (b) 验证损失

图 6：*CLRNet 的训练与验证损失*：CLRNet 在 DeepFake (DF)、FaceSwap(FS)、Face2Face (F2F)、NeuralTextures (NT) 和 DeepFakeDetection (DFD) 数据集上的训练和验证损失。训练和验证的损失递减过程非常相似，这表明我们的模型学习准确，没有过度拟合训练数据。

*3.5.2 ShallowNet.*Tariq *等人*[39]的研究表明，ShallowNet[38]在检测计算机生成的图像方面达到了很高的精度。我们使用 Python v3.6.8 和 TensorFlow v1.14.0 [1]，并使用 Keras v2.2.4 [8]，开发了 ShallowNet。

*3.5.3 FaceForensics++ (FF++)。*Rössler *等人*[35] 使用改进版的 Xception 网络来检测 DeepFake、FaceSwap、Face2Face 和 NeuralTextures。我们直接使用 FaceForensics++ [35]的结果，因为他们使用了相同的数据集。

*3.5.4 带有双向 RNN 的 DenseNet。*Sabir *等人*[36] 使用带有双向 RNN 的 DenseNet 在 DeepFake、FaceSwap 和 Face2Face 数据集上取得了很高的准确率。与我们的 CLRNet 类似，这项工作也使用五个连续帧来训练和测试模型。我们直接使用 Sabir *等人的*结果[36]，因为他们使用了相同的数据集。

*3.5.5 Forensics Transfer (FT)。*Cozzolino *等人*[10]利用基于自动编码器的方法，开发了一种用于域适应的弱监督方法。他们将潜在空间分为真实和虚假两部分，从而在 Face2Face 和 FaceSwap 数据集上实现了更高的检测准确率。为了实现 ForensicTransfer 自动编码器[10]，我们在 Python v3.6.8 上使用了 PyTorch v1.1.0 [32]。

4 结果

我们进行了大量实验来评估和比较 CLRNet 和基准方法的性能。本文只讨论最重要的实验及其结果。下文将详细讨论不同实验的结果。

4.1 CLRNet 的学习能力

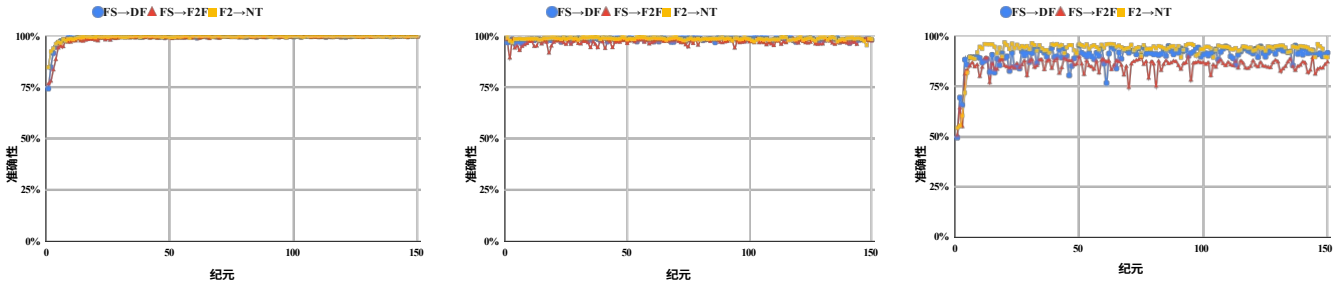
图 6 显示了 CLRNet 在不同数据集（DF、FS、F2F、NT 和 DFD）上的训练和验证损失。我们可以观察到，训练损失和验证损失以类似的方式逐渐下降，这表明我们的 CLRNet 模型没有过度拟合训练数据集，并且从训练集中学习到了可区分真假图像的特征。如图 6 所示，DFD 的验证损失有轻微波动，但其他数据集则没有这种情况。我们认为，这是由于 DFD 如表 1 所示，用于 DFD 训练的数据集（250 个视频）与其他数据集（750 个视频）相比规模较小。另一个原因是 DFD 视频的动态环境增加了模型学习的难度。不过，即使数据量较小，我们的 CLRNet 模型也能从 DFD 数据集中学习，并取得了 96.00% 的 F1 分数，如表 2 所示。

4.2 基础数据集的性能

如表 2 所示，我们在基本训练数据集（DF 或 FS）上训练了 CLRNet 和其他基线模型。然后，我们比较了训练模型在基础数据集测试集上的性能，并评估了其他数据集测试集的零点性能。基线方法（ShallowNet、Xception 和 FT）和 CLRNet 是在数据集的图像集上训练和评估的。不过，FF++ [35] 和 Sabir *等人* [36] 的结果直接取自他们的工作。表 2 报告了所有结果。从表 2 中可以看出，在 DeepFake 基础数据集上进行训练时，FT（99.35%）略微领先 CLRNet（99.02%）。我们的 Xception（86.00%）和 ShallowNet（56.65%）的虚构实现性能不佳。不过，FF++ 的改进版 Xception （96.36%）和 Sabir *等人的* DenseNet *（*96.90%）比 ShallowNet（56.65%）和 vanilla Xception（86.00%）表现更好。所有方法的零镜头性能都相对较低（低于 85%）：FT 在 F2F（74.12%）和 NT（83.54%）上表现最佳，而 CLRNet 在 DFD（60.38%）上表现最佳。如表 2 所示，当使用 FaceSwap（FS）基础数据集进行训练时，与最佳基线方法 FT（FS：96.17%）相比，CLRNet（FS：98.05%）表现最佳。与 DF 基础数据集类似，Xception（FS：85.37%）、FF++（FS：90.29%）和 Sabir el al.（FS：94.35%）表现尚可，ShallowNet 表现最差（FS：52.75%）。我们还在 NeuralTextures（NT）和 DeepFakeDetection（DFD）基础数据集上评估了 CLRNet 方法。如表 2 所示，CLRNet 在这两个数据集上都获得了非常高的 F1 分数（NT：99.50%，DFD：96.00%）。从表 2 中我们可以看出，在零点性能方面没有最佳方法。因此，我们将在接下来的实验中探索迁移学习。

4.3 有源和目标的迁移学习

在我们的初步实验中，我们发现当模型仅使用目标数据集进行迁移学习训练时，它在源数据集上的性能会下降（低至 50%），这对于构建通用的深度假货检测器来说并不理想。因此，我们将少量源数据集（10 个视频）和目标数据集（10 个视频）结合起来进行迁移学习，以便模型也能记住源数据集的特征。图 7 显示了训练准确率（源 + 目标）、源数据集的验证准确率（FS）和目标数据集的验证准确率（DF 或 F2F 或 NT）。从图 7 中可以看出，源数据集（FS）的验证准确率略有下降（1%~2%），而目标数据集（DF 或 F2F 或 NT）的验证准确率随着时间的推移不断提高（达到 90%）。这种技术使 CLRNet 在源数据集和目标数据集的迁移学习方面都取得了很高的性能。下一节，我们将讨论迁移学习的结果。



(a) ***训练（源 + 目标）*****：我们从源数据集和目标数据集各使用了十段视频。即使数据量很小，CLRNet 也显示出很高的学习能力。**

(b) ***验证（源）*****：在转发学习过程中，我们从源数据集中提供了十段视频，从而实现了源数据集的高准确性。**

(c) ***验证（目标）*****：我们在使用 CLRNet 的目标数据集上只使用了十段视频，也取得了相当高的准确率。**

图 7：*CLRNet 的迁移学习准确率*：(a) 转移学习 (TL) 训练准确率与 (b) 源数据集验证准确率和 (c) 目标数据集验证准确率的比较。箭头"→"代表从源数据集到目标数据集的迁移学习。

4.4 迁移学习成绩

对于迁移学习，我们只使用了 CLRNet 和基础数据集上表现最好的基线模型，即 FT。Cozzolino *等人*[10]仅对 FF++ 数据集[35]进行了 Face2Face（源）到 FaceSwap（目标）的实验。我们用 FT 进行了大量实验，并与我们的 CLRNet 模型进行了比较。我们针对迁移学习（TL）任务进行了三种不同类型的实验，每种实验由多个子实验组成。下文将详细介绍这些实验以及 CLRNet 和 FT 的性能评估。

*4.4.1 Exp 1.单源到单目标。*在这个实验中，我们使用一个源数据集（FS、F2F、NT、DFD 或 DF）和一个目标数据集（FS、F2F、NT、DFD 或 DF）进行迁移学习，并比较了 FT 和 CLRNet 的性能。由于篇幅有限，我们在表 3 中部分报告了这一实验。在实验 1 中，我们将 DeepFake（DF）设为源，将 FaceSwap（FS）设为目标。如表 2 和表 3 所示，目标（FS）的 CLRNet 准确率从 50.00% 提高到 83.08%，源（DF）的 CLRNet 准确率从 99.02% 下降到 90.95%。相比之下，FT 的准确率从 50.00%（FS）和 99.35%（DF）分别下降到 47.72%（FS）和 62.59%（DF）。

如表 2 和表 3 所示。同样，当以 Face2Face（F2F）为目标时，CLRNet 对目标的准确率从 53.73% 提高到 88.35%，而 FT 的准确率只提高了 5%，从 74.12% 提高到 79.31%。这种低性能表明了 FT 在泛化方面的局限性。在实验 2 中，我们将 FaceSwap（FS）设为源，目标数据集（DF 和 F2F）的性能比实验 1 中的更好，有利于 CLR- Net。当以 DeepFake（DF）为目标时，CLRNet（FS：96.33%，DF：92.47%）的性能比 FT（FS：44.93%，DF：76.94%）分别高出 50%（FS）和 16%（DF）。同样，当以 Face2Face（F2F）为目标时，CLRNet（FS：93.93%，F2F：87.48%）优于 FT（FS：55.74%，F2F：53.73%），如表 3 所示。我们还对 CLRNet 进行了实验，将目标设置为 NeuralTextures（NT）或 DeepFakeDetection（DFD）。如表 2 和表 3 所示，经过迁移学习后，源（FS）在 NT 上的性能从 98.05% 提高到 98.12%，目标（NT）从 53.33% 提高到 95.50%。最后，对于 DFD，我们

观察到类似的趋势，实现了从 49.13% 到 88.88% 的增长。

如表 2 和表 3 所示，根据我们的实验，CLR-Net 对目标的准确度总是比其零点性能提高，而对源的准确度则略有下降或有时提高。另一方面，FT 的性能非常不稳定，通常比单一数据集的性能更差，这表明 FT 有很大的局限性；而 CLRNet 克服了这一局限性，取得了更好的性能。

*4.4.2 Exp 2.多源到单目标。*在这个实验中，我们首先用两个不同的源数据集训练模型，然后对一个新的目标数据集进行迁移学习。在表 3 中，我们报告了本实验中 FT 和 CLRNet 表现最好的模型，即 FS+DF（源）和 F2F（目标）。与 FT（FS：45.17%，DF：64.08%）相比，CLRNet（FS：94.37%，DF：92.15%）在源数据集上的准确率明显更高。同样，在目标数据集上，CLRNet（F2F：86.22%）也优于 FT（F2F：55.98%），如表 3 所示。在 NT（CLRNet：79.75%，FT：62.09%）和 DFD（CLRNet：63.12%，FT：51.38%）上，CLRNet 的零点学习性能也超过了 FT。该实验还验证了 CLRNet 在深度伪造检测泛化方面优于 FT。

*4.4.3 实验 3：从单一来源到多目标。*在本实验中，我们将测试 CLRNet 模型的通用性，即其在仅有少量数据的情况下同时检测多种深度伪造类型的性能。在本实验中，我们首先在源数据集（FS）上训练了 CLRNet 模型，然后进行了三种类型的 实验。在实验 1 中，我们将目标设为 DF+F2F，并评估了迁移学习后模型的准确性。从表 3 中，我们可以看到源数据集（FS：94.55%）和目标数据集（DF：91.77%，F2F：87.75%）的表现。在实验 2 中，我们将目标设定为 DF+F2F+NT。从表 3 中可以看出，源（FS：94.35%）和目标（DF：90.67%，F2F：85.78%，NT：91.47%）的准确率。在本实验中，我们将目标设为 DF+F2F+NT+DFD。在表 3 中，我们可以看到源的准确率（FS：93.70%）和目标的准确率（F2F：85.78%）。

表 2：*基础数据集和零点性能*：该表显示了 CLRNet 与五种不同基线模型的性能比较。首先，我们在基础数据集上对模型进行了训练。然后，我们在基础数据集（灰色突出显示）和其他数据集（零点）上测试了所有模型。除了一个基础数据集之外，我们的 CLRNet 在其他所有基础数据集上的性能都是最高的。不过，就零拍摄性能而言，CLRNet 在 FS 和 FT 上的表现也是最好的
与 DF 和 F2F 一样。‡"代表我们对
方法。

方法	基地 数据集	DF (%)	FS (%)	F2F (%)	NT (%)	DFD (%)
浅网‡	DF	56.65	**50.32**	54.15	35.13	41.23
Xception		86.00	49.78	57.26	61.49	**68.10**
FF++ [35]		96.36	-	-	-	-
萨比尔等人 [36]		96.90	-	-	-	-
FT‡		**99.35**	50.00	**74.12**	**83.54**	48.45
CLRNet （我们的）		99.02	50.00	53.73	69.75	60.38
浅网‡	FS	47.09	52.75	47.50	46.96	**49.58**
Xception		**49.49**	85.37	**56.61**	52.05	48.08
FF++ [35]		-	90.29	-	-	-
萨比尔等人 [36]		-	94.35	-	-	-
FT‡		48.86	96.17	54.39	47.45	36.66
CLRNet （我们的）		48.53	**98.05**	50.15	**53.33**	49.13
FF++ [35]	NT	-	-	-	80.67	-
CLRNet （我们的）		**50.12**	**49.93**	**49.80**	**99.50**	**50.00**
CLRNet （我们的）	DFD	**65.08**	**51.92**	**60.32**	**63.10**	**96.00**

和目标（DF：91.23%；F2F：87.50%；NT：91.30%；DFD：87.13%）。基于这些实验，我们得出以下结论1）随着目标数据集数量的增加，源数据集的准确率会略有下降（1%~2%）；2）CLRNet 在不同的 deepfake 数据集上具有良好的泛化能力，因为在我们所有的实验中，CLRNet 的性能都优于相同数据集上的 zero-shot 性能。

5 结论

我们引入了 CLRNet，并将其成功应用于多种深度伪造视频的检测。我们的模型不使用单帧，而是使用视频中的连续帧序列作为输入，这有助于 CLRNet 模型捕捉和整合时间信息，并检测连续帧之间存在的伪影。通过 5 个以上不同的实验，我们证明了 CLRNet 模型优于之前的基线和最先进的方法。总之，我们提出了一个更具通用性、检测性能更佳的模型，从而解决了以往最先进方法的不足之处。从这项工作中，我们希望我们的研究是通往开发更通用的深度赝品检测器的垫脚石，未来的工作将继续挑战和改进现有的深度赝品检测方法，使其更具通用性和普遍性。

表 3：*迁移学习性能*：该表显示了我们的 CLRNet 模型与之前实验中的最佳基线方法（即 FT）的性能对比。在本实验中，我们首先在源数据集上训练模型，然后使用由 10 个视频组成的小型目标数据集（灰色突出显示）进行迁移学习。之后，我们在所有数据集上进行测试。我们进行了 10 次实验，分别属于 3 种不同类型：1) 单源到单目标；2) 多源到单目标；3) 多源到单目标；4) 多源到单目标。和 3）单源到多目标。在所有情况下，我们的 CLRNet 模型都表现最佳。‡"表示我们对该方法的实施。

方法	资料来源	目标	DF (%)	FS (%)	F2F (%)	NT (%)	DFD (%)
从单一来源到单一目标							
FT‡	DF	FS	62.59	47.72	**61.34**	**68.51**	51.38
CLRNet			**90.95**	**83.08**	48.68	65.00	**53.87**
FT‡		F2F	83.70	**54.92**	79.31	**82.24**	54.92
CLRNet			**97.18**	49.28	**88.35**	78.05	**68.13**
FT‡	FS	DF	76.94	44.93	**66.42**	70.53	49.78
CLRNet			**92.47**	**96.33**	65.58	**75.80**	**59.13**
FT‡		F2F	56.06	55.74	53.73	55.38	47.63
CLRNet			**79.87**	**93.93**	**87.48**	**78.00**	**52.50**
FT‡		NT	-	-	-	-	-
CLRNet			51.82	98.12	50.90	95.50	49.13
FT‡		DFD	-	-	-	-	-
CLRNet			70.97	96.15	51.85	77.02	88.88
多源到单目标							
FT‡	FS+DF	F2F	64.08	45.17	55.98	62.09	51.38
CLRNet			**92.15**	**94.37**	**86.22**	**79.75**	**63.12**
从单一来源到多目标							
CLRNet（最佳）	FS	DF+F2F	**91.77**	**94.55**	**87.75**	85.12	69.13
		DF+F2F +NT	90.67	94.35	85.78	**91.47**	69.50
		DF+F2F +NT+DFD	91.23	93.70	87.50	91.30	**87.13**

参考文献

[1] Martín Abadi、Paul Barham、Jianmin Chen、Zhifeng Chen、Andy Davis、Jeffrey Dean、Matthieu Devin、Sanjay Ghemawat、Geoffrey Irving、Michael Isard 等人，2016 年。Tensorflow：大规模机器学习系统。*第 12 届{USENIX}大会操作系统设计与实现研讨会（{OSDI} 16）*。265-283.

[2] Shruti Agarwal 和 Hany Farid.2017.从 JPEG 凹点进行照片取证。在 *2017 IEEE 信息取证与安全研讨会（WIFS）*。IEEE，1-6。

[3] Shruti Agarwal, Hany Farid, Yuming Gu, Mingming He, Koki Nagano, and Hao Li.2019.保护世界领袖免受深度伪造。In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*.38-45.

[4] Jawadul H Bappy、Amit K Roy-Chowdhury、Jason Bunk、Lakshmanan Nataraj 和 BS Manjunath。2017.利用空间结构定位操纵图像区域。*电气和电子工程师学会计算机视觉国际会议论文集*》。4970-4979.

[5] Belhassen Bayar and Matthew C Stamm.2016.使用新卷积层的通用图像操作检测深度学习方法。*第四届 ACM 信息隐藏与多媒体安全研讨会论文集*》。ACM, 5-10.

[6] 深度伪造检测挑战赛。2019.Deepfake Detection Challenge：//www.kaggle.com/c/deepfake-detection-challenge. 访问时间：2020-02-12。

[7] 弗朗索瓦-乔莱2017.Xception：深度学习与深度可分离构图。In *Proceedings of the IEEE conference on computer vision and pattern recognition*.1251-1258.

[8] François Chollet 等人，2015 年。Keras。https://keras.io。

[9] Davide Cozzolino、Giovanni Poggi 和 Luisa Verdoliva。2017.将基于残差的局部描述符重塑为卷积神经网络：应用于图像伪造检测。*第五届 ACM 信息隐藏 和多媒体安全研讨会论文集》，ACM, 159-164.*ACM，159-164。

[10] Davide Cozzolino、Justus Thies、Andreas Rössler、Christian Riess、Matthias Nießner 和 Luisa Verdoliva。2018.Forensictransfer：用于伪造检测的弱监督域 适应。*arXiv preprint arXiv:1812.02510* (2018)。

[11] 阿德里安-克罗夫特2019.从色情到诈骗，深度伪造正在成为一个大骗局--这让商业领袖和法律界人士感到不安。https://fortune.com/2019/10/07/porn-to-scams-deepfakes-big-racket- 让商业领袖和法律界人士感到不安。访问时间：2020-02-11。

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei.2009.Imagenet：大规模分层图像数据库。*2009 IEEE 计算机 视觉与模式识别会议*。IEEE，248-255。

[13] EJ Dickson.2019.Deepfake Porn Is Still a Threat, Particularly for K-Pop Stars. https://www.rollingstone.com/culture/culture-news/deepfakes-nonconsensual-porn-study-kpop-895605.访问时间：2020-02-11.

[14] 夏洛特-爱德华兹2019.专家称，制作 deepfake porn 很快就能像使用 Instagram 滤镜一样简单。https://www.thesun.co.uk/tech/9800017/ deepfake-porn-soon-easy。访问时间：2020-02-11.

[15] FaceSwapDevs.[n.d.].Deepfakes_faceswap - GitHub Repository. https://github.com/deepfakes/faceswap.Accessed: 2019-11-05.

[16] Pasquale Ferrara、Tiziano Bianchi、Alessia De Rosa 和 Alessandro Piva。2012.通过细粒度 CFA 伪影分析进行图像伪造定位。*IEEE Trans- actions on Information Forensics and Security* 7, 5 (2012), 1566-1577.

[17] 亚历克斯-格雷夫斯2013.用递归神经网络生成序列。*arXiv preprint arXiv:1308.0850* (2013)。

[18] David Güera and Edward J Delp.2018.使用递归神经网络进行深度伪造视频检测。*2018 年第 15 届 IEEE 高级视频 和基于信号的监控（AVSS）国际会议*。IEEE，1-6。

[19] 何开明、张翔宇、任绍清、孙健。2016.图像识别的深度残差学习。In *Proceedings of the IEEE conference on computer vision and pattern recognition*.770-778.

[20] Minyoung Huh, Andrew Liu, Andrew Owens, and Alexei A Efros.2018.打击假新闻：通过学习的自一致性进行图像拼接检测。*欧洲计算机视觉会议（ECCV）论文集* 。101-117.

[21] Hyeonseong Jeon, Youngoh Bang, and Simon S Woo.2019.FakeTalkerDetect：使用高度不平衡数据集进行有效实用的现实神经话头检测。*电气和电子工程师学会计算机国际会议论文集 Vision Workshops*.0-0.

[22] Hyeonseong Jeon, Youngoh Bang, and Simon S Woo.2020.FDFtNet：FDFtNet: Facing Off Fake Images using Fake Detection Fine-tuning Network. *ArXiv preprint arXiv:2001.01265* (2020).

[23] Michael Kan.大多数人工智能生成的在线深度伪造视频都是色情。https://www.pcmag.com/news/371193/most-ai-generated-deepfake-videos-online-are-porn。访问时间：2020-02-11.

[24] Tero Karras、Timo Aila、Samuli Laine 和 Jaakko Lehtinen。2017.为提高质量、稳定性和变异性而进行的甘蔗逐步生长。*arXiv 预印本 arXiv:1710.10196* (2017)。

[25] 马雷克-科瓦尔斯基2016.FaceSwap - GitHub Repository. https://github.com/MarekKowalski/FaceSwap。访问时间：2020-02-12。

[26] Yuezun Li, Ming-Ching Chang, and Siwei Lyu.2018.In ictu oculi: Exposing ai created fake videos by detecting eye blinking.In *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*.IEEE, 1-7.

[27] Yaqi Liu, Qingxiao Guan, Xianfeng Zhao, and Yun Cao.2018.基于多尺度卷积神经网络的图像伪造定位。*第六届 ACM 信息隐藏与多媒体安全研讨会论文集》* 。ACM， 85-90。

[28] Falko Matern、Christian Riess 和 Marc Stamminger。2019.利用视觉伪影揭露深度伪造和人脸操纵。In *2019 IEEE Winter Appli- cations of Computer Vision Workshops (WACVW)*.IEEE，83-92。

[29] Ghazal Mazaheri, Niluthpol Chowdhury Mithun, Jawadul H Bappy, and Amit K Roy-Chowdhury.2019.用于图像操作定位的跳接连接架构。*电气和电子工程师学会计算机视觉和 模式识别研讨会论文集》*。119-129.

[30] 伊万 - 梅塔2019. 一项新研究称近 96 个 deepfake 视频是色情视频。https://thenextweb.com/apps/2019/10/07/a-new-study-says-nearly-96-of-deepfake-videos-are-porn。访问时间：2020-02-11.

[31] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu.2019.空间自适应归一化的语义图像合成。In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.2337-2346.

[32] Adam Paszke、Sam Gross、Soumith Chintala、Gregory Chanan、Edward Yang、Zachary DeVito、Zeming Lin、Alban Desmaison、Luca Antiga 和 Adam Lerer。2017.PyTorch 中的自动差异化。在 *NIPS Autodiff 研讨会上*。

[33] Yuan Rao and Jiangqun Ni.2016.图像拼接和复制移动伪造检测的深度学习方法。In *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*.IEEE，1-6。

[34] Andreas Rössler、Davide Cozzolino、Luisa Verdoliva、Christian Riess、Justus Thies 和 Matthias Nießner.2018.Faceforensics：用于人脸伪造 检测的大规模视频数据集。*arXiv preprint arXiv:1803.09179* (2018)。

[35] Andreas Rössler、Davide Cozzolino、Luisa Verdoliva、Christian Riess、Justus Thies 和 Matthias Nießner.2019.FaceForensics++：学习检测操纵的 面部图像。In *ICCV 2019*.

[36] Ekraam Sabir, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, and Prem Natarajan.2019.用于视频中人脸操纵 检测的循环卷积策略。*Interfaces (GUI)* 3 (2019), 1.

[37] Ronald Salloum, Yuzhuo Ren, and C-C Jay Kuo.2018.使用多任务全卷积网络（MFCN）进行图像拼接定位。*视觉 Communication and Image Representation* 51 (2018)，201-209.

[38] Shahroz Tariq、Sangyup Lee、Hoyoung Kim、Youjin Shin 和 Simon S Woo。2018.在野外检测机器和人类创建的伪造人脸图像。*第二届多媒体隐私与安全国际研讨会论文集》*。ACM，81-87。

[39] Shahroz Tariq、Sangyup Lee、Hoyoung Kim、Youjin Shin 和 Simon S Woo。2019.GAN是敌是友？*第 34 届 ACM/SIGAPP 应用计算研讨会论文集*》。ACM，1296-1303。

[40] Justus Thies, Michael Zollhöfer, and Matthias Nießner.2019.延迟神经纹理：使用神经纹理的图像合成。*arXiv preprint arXiv:1904.12356* (2019)。

[41] Justus Thies、Michael Zollhöfer、Marc Stamminger、Christian Theobalt 和 Matthias Nießner.2018.Face2Face：实时人脸捕捉和 RGB 视频重演。*Commun。ACM* 62, 1 (Dec. 2018), 96-104. https://doi.org/10.1145/3292039

[42] 亚历山大-泽维尔 2019.ConvLSTM 简介。https://medium.com/ neuronio/an-introduction-to-convlstm-55c9025563a7 访问时间：2019-11-05。

[43] Shi Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo.2015.卷积 LSTM 网络：降水预报的机器学习方法。In *Advances in neural information processing systems*.802-810.

[44] Xin Yang, Yuezun Li, and Siwei Lyu.2019.利用不一致的头部姿势揭露深度伪造。In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.IEEE, 8261-8265.

[45] Ido Yerushalmy and Hagit Hel-Or.2011.基于镜头和传感器像差的数字图像伪造检测。*国际计算机视觉杂志》* 92，1（2011），71-91。

[46] 张凯鹏、张展鹏、李志峰、乔宇。2016.使用多任务级联卷积网络进行联合人脸检测和对齐。*IEEE Signal Processing Letters* 23, 10 (2016), 1499-1503.

[47] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis.2018.学习图像操作检测的丰富特征。*电气和电子工程师学会计算机视觉与模式识别会议论文集》*。1053-1061.

A 关于重现性的附录

本节介绍重现 CLRNet 和所有基线方法实验结果所需的全部必要信息。在此，我们尽力涵盖所有内容，以帮助实现可重复性。

A.1 硬件和软件配置

我们使用英特尔（R）至强（R）Silver 4114 CPU（主频为 2.20GHz）、256.0GB 内存和英伟达（NVIDIA）GeForce Titan RTX。在 Python v3.7.5 上使用了以下软件包版本：TensorFlow-gpu：v1.13.1；Keras-applications：v1.0.8；Keras-preprocessing：v1.1.0；cudatoolkit：v10.0.130；numpy：v1.17.4；Pandas：v0.25.3；Opencv：3.4.2；scikit- learn：v0.21.3；PyTorch：v1.1.0。

A.2 Deepfake 视频数据集

我们从 FaceForensics++ GitHub 存储库[1] 下载了数据集（DeepFake、Face2Face、FaceSwap、Neu- ralTextures 和 DeepFakeDetection），用于训练和评估基线模型和 CLRNet 模型。

A.3 实施 CLRNet

我们使用 Python 上的 TensorFlow 和 Keras 实现了 CLRNet 模型。Keras 库中提供了模型中使用的所有层。CLRNet 架构的详细视图见表 6。我们使用 Keras 惯例来命名层。我们模型的输入形状是（视频、样本、行、列、通道）。Keras 中的 ImageDataGenerator 可以加载（样本、行、列、通道）形式的数据。因此，我们无法使用 Keras 库提供的 ImageDataGenerator。因此，我们构建了自己的 VideoDataGenerator，其功能与 ImageDataGenerator 非常相似。不过，由于我们计划提供一组 5 个连续帧的输入，因此还必须实现一些额外的功能。所有这些都包含在我们的源代码中，我们计划将其与本文一同发布。

A.4 基线模型的实施

我们将 CLRNet 模型与五个基准模型（Xcep-tion、ShallowNet、FF++、DenseNet with Bidirectional RNN 和 FT）进行了比较。下面，我们将解释这些基线模型是如何实现的。

A.4.1 Xception 我们使用了 Xception 的 Keras 实现，该实现已在 ImageNet 数据集上进行了预训练。

A.4.2 ShallowNet. 我们根据 Tariq 等人提供的规范实现了 ShallowNetV3 [38]。Shal- lowNetV3 的架构如表 4 所示。ShallowNetV3 由 4 个块、33 层组成。表 4 中的一行对应一个区块。我们使用 Keras 和 TensorFlow 作为后端实现了这一架构。

A.4.3 ForensicTransfer (FT)。 我们根据 Cozzolino 等人[10] 提供的规范实现了 ShallowNetV3，取证转移（FT）模型的架构如表 5 所示。我们使用 PyTorch v.1.2.0 实现了 FT 模型。我们尽力模仿基线的原始性能。

[1] https://github.com/ondyari/FaceForensics

A.5 为 CLRNet 准备数据

在准备数据集时，我们从每段视频中随机抽取 5 个连续帧作为一个样本。我们从每段视频中提取了 16 个样本。用于训练的视频帧未用于验证或测试。我们使用 MTCNN [46] 对帧中的人脸进行了裁剪。我们使用我们的 VideoDataGen- erator 加载这些样本，并将其输入 CLRNet。

A.6 数据扩充

在 Keras 中，数据增强功能可与图像数据生成器一起使用。由于我们已经实现了 VideoDataGenerator，因此还必须实现我们自己的 DataAugumentor。因此，我们使用了 Keras ImageDataGenerator 接口，并将其扩展为 VideoDataGenerator。我们用于数据增强的变换设置如下：rotation_range=30; brightness_range = [0.7,1.0]; channel_shift_range=50.0; zoom_range=0.2; horizon-tal_flip=True; fill_mode='nearest'。

A.7 模型培训

我们使用的亚当优化器设置如下：lr = 0.00005; beta_1 = 0.9; beta_2 = 0.999; epsilon = None; decay = 0.0; amsgrad = 假。对于损失函数，我们使用了二元交叉熵。在每个实验中，我们都对 CLRNet 和基线模型进行了 100 次历时训练。基于验证损失的最佳历时用于测试数据集的评估。我们在训练、验证和测试数据集中保持了相同数量的真实样本和虚假样本。在基线方法的训练中，我们使用了与原始论文中相同的训练和测试方法。

A.8 迁移学习

在迁移学习过程中，我们冻结了 CLRNet 模型的前 120 层。在 Keras 中，当涉及迁移学习时，批量归一化层存在一个小问题。我们使用 Vasilis Vryniotis[2] 和 Oleg Gusev[3] 提供的解决方案解决了这个问题。在迁移学习方面，我们每次实验使用 150 个历元，其他条件保持不变。用于转移特定目标的数据集数量为 10 个视频，如果有两个目标，则每个目标使用 10 个视频。

A.9 源代码

我们计划发布 CLRNet 的实现、我们的实验、ShallowNet 和 ForensicsTransfer (FT) 的源代码。如果我们的论文被接受，我们将清理代码并分享到 GitHub。源代码还包含所有实验的 Juypter Notebooks 及其结果。

[2] http://blog.datumbox.com/the-batch-normalization-layer-of-keras-is-broken/

[3] https://github.com/keras-team/keras/pull/9965#issuecomment-549064001

表 4：ShallowNet 架构。我们使用以下 ShallowNetV3 架构来开发此基线模型。每一行代表架构中的一个区块。每个 Conv2D 层都使用了 0.0001 的 L2 内核正则化器。

ShallowNetV3
Conv2D - ReLU - Dropout - Conv2D - ReLU - Dropout - Conv2D - ReLU - MaxPooling - Dropout
Conv2D - ReLU - Dropout - Conv2D - ReLU - Dropout - Conv2D - ReLU - MaxPooling - Dropout
Conv2D - ReLU - Dropout - Conv2D - ReLU - Dropout
扁平化 - 密集 - ReLU - 批量归一化 - 剔除 - 密集 - 西格莫德

表 5：ForensicsTransfer（FT）架构。我们使用以下架构来开发 FT 基线模型。编码器部分的每个 Conv2d 的步长为 2，而解码器部分的每个 Conv2d 的步长为 1。

法医转岗（全职）	
编码器	Conv2d - ReLU - Conv2d - BatchNorm2d - ReLU - Conv2d - BatchNorm2d - ReLU - Conv2d - BatchNorm2d - ReLU - Conv2d - BatchNorm2d - LReLU
解码器	Upsample - Conv2d - BatchNorm2d - ReLU - Upsample - Conv2d - BatchNorm2d - ReLU - Upsample - Conv2d - BatchNorm2d - ReLU - Upsample - Conv2d - BatchNorm2d - ReLU - ConvTranspose2d - Tanh

表 6：这是我们在本文中使用的 CLRNet 模型的详细架构。Keras 库中提供了所有这些层。图像的输入大小为 240x240。

基于卷积 LSTM 的残差网络（CLRNet）
ConvLSTM2D - BatchNorm - ReLU - MaxPooling3D
ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ConvLSTM2D - ReLU - BatchNorm - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ConvLSTM2D - ReLU - BatchNorm - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ConvLSTM2D - ReLU - BatchNorm - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ConvLSTM2D - ReLU - BatchNorm - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - Dropout - BatchNorm - ReLU - ConvLSTM2D - **Add**
BatchNorm - GlobalAveragePooling3D - Dropout - Dense