Future Generation Computer Systems 112 (2020) 945–955

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

PPCensor: Architecture for real-time pornography detection in video streaming

Jackson Mallmann [a,b], Altair Olivo Santin [b,*], Eduardo Kugler Viegas [b], Roger Robson dos Santos [b], Jhonatan Geremias [b]

[a] Federal Institute Catarinense, Brazil
[b] Pontifical Catholic University of Paraná (PUCPR), Brazil

ARTICLE INFO

Article history:
Received 22 November 2019
Received in revised form 14 May 2020
Accepted 11 June 2020
Available online 19 June 2020

Keywords:
Convolutional neural networks
Object-oriented private parts detection
Private parts object dataset
Video-oriented streaming proxy
Near real-time detection
Resource-Constrained Devices
Private Parts Censor

ABSTRACT

Convolutional neural network (CNN) models are typically composed of several gigabytes of data, requiring dedicated hardware and significant processing capabilities for proper handling. In addition, video-detection tasks are typically performed offline, and each video frame is analyzed individually, meaning that the video's categorization (class assignment) as normal or pornographic is only complete after all the video frames have been evaluated. This paper proposes the Private Parts Censor (PPCensor), a CNN-based architecture for transparent and near real-time detection and obfuscation of pornographic video frame regions. Our contribution is two-fold. First, the proposed architecture is the first that addresses the detection of pornographic content as an object detection problem. The objective is to apply user-friendly content filtering such that an inevitable false positive will obfuscate only regions (objects) within the video frames instead of blocking the entire video. Second, the PPCensor architecture is deployed on dedicated hardware, and real-time detection is deployed using a video-oriented streaming proxy. If a pornographic video frame is identified in the video, the system can hide pornographic content (private parts) in real time without user interaction or additional processing on the user's device. Based on more than 50,000 objects labeled manually, the evaluation results show that the PPCensor is capable of detecting private parts in near real time for video streaming. Compared to cutting-edge CNN architectures for image classification, PPCensor achieved similar results, but operated in real time. In addition, when deployed on a desktop computer, PPCensor handled up to 35 simultaneous connections without the need for additional processing on the end-user device.

1. Introduction

The spread of pornographic videos on the internet, including those with explicitly pornographic content, has increased significantly in recent years [1]. For example, in 2018, one of the most visited pornographic sites received more than 4 million daily video uploads [2]. Exposure to this undesirable content, for example, by underage people without parental consent, can cause embarrassment or even psychological trauma [3]. Concerns regarding this kind of content are manifested by more than 71% of those responsible for underage children who are browsing the web [4].

Convolutional neural networks (CNNs) have, in general, demonstrated promising results in detecting pornographic video content [5–9]. For CNN analysis, a video is divided into a set of video frames (images), and each image is classified individually [10]. As each video consists of several frames and the typical frame rate is 23.97 frames per second (FPS) [11], current detection techniques cannot be used in real time, especially given that they require the entire set of video frames.

In real-world applications, videos are often streamed [12]. For instance, streaming pornographic websites (URL), such as LiveJasmin, Pornhub, XVideos, and xHamster, were included in the 100 most popular websites published by Alexa Internet in August 2019. In addition, because each video frame is classified individually, a single video can contain multiple frames labeled as pornographic, even if the video frame sequence does not contain inappropriate content for an underaged person.

A CNN-based detection approach may misclassify some video frames, and consequently, classify the video incorrectly [13]. Therefore, CNN-based video filtering can result in a high false positive (FP) rate, a situation in which several normal video frames are incorrectly classified as pornographic, as well as false negatives (FN), pornographic video frame deemed as normal.

* Corresponding author.
E-mail addresses: jackson.mallmann@ifc.edu.br, jackson.mallmann@pucpr.edu.br (J. Mallmann), altair.santin@pucpr.br (A.O. Santin), eduardo.viegas@pucpr.br (E.K. Viegas), robson.roger@ppgia.pucpr.br (R.R. dos Santos), j.geremias@ppgia.pucpr.br (J. Geremias).

https://doi.org/10.1016/j.future.2020.06.017
0167-739X/

Moreover, CNN-based detection techniques often require a significant amount of memory and CPU/GPU capabilities [14]. One widely used CNN architecture can require up to 16 GB of RAM while processing only 4.26 FPS on a high-end GPU [15].

CNN deployment on an end-user device has become challenging, considering that the available detection approaches ignore the limited resources available for processing on end-user devices [16], including smartphones and tablets. As a result, current video filtering techniques based on the latest generation CNNs are unsuitable for deployment in such environments because their high computational demands typically require expensive and dedicated high-end hardware [13].

This paper presents PPCensor, an architecture for real-time pornographic object detection of video streaming frames. Our proposed technique is based on two main insights. First, to provide reliable real-time detection of private parts in video frames, our approach is the first that treats pornographic content as an object detection problem. The goal is to hide only the inappropriate areas within a set of frames instead of blocking access to the entire video sequence, as is often done in the literature. Consequently, PPCensor is less harsh than other approaches as the user retains access to the full video with potentially inappropriate objects being hidden. In addition, even if an FP occurs, the user can watch the affected frames with blurred (obfuscated) regions. Therefore, the user experience when using PPCensor is not significantly degraded owing to an FP. Second, to enable usage in production environments (real-world use), PPCensor is implemented as a network proxy server for video streams. The main advantage is the transparent, remote classification of the video streaming frames, i.e., running the CNN detection through a proxy instead of on the end-user device. PPCensor is deployed in dedicated hardware, and it evaluates the video content of all queried video URLs in real time. If pornographic video content is found, it may block access or hide inappropriate objects inside a frame without user intervention. PPCensor only requires the user to apply a proxy server to enable the device-agnostic filtering of inappropriate content.

The contributions of this study have been summarized below.

- We provide the first publicly available dataset with annotated private parts of a human body (objects), the Private Parts Object Dataset (PPO). This dataset was created by the manual inspection of pornographic video frames, resulting in 52,215 annotated objects. The dataset provides a benchmark for the research community that is building novel object-based pornographic detection approaches aimed at user-friendly detection and obfuscation of the private part.
- We present a novel technique that considers pornography detection as an object detection problem. The main advantage of the object detection approach is that we can hide (obfuscate) only the private parts (objects) within a video frame. Without using object detection, it would be necessary to remove or block entire video frames from the video stream to hide potential pornographic content, causing visual discontinuities in the media and degrading the user experience. Our proposed model achieves a low error rate (observed in the low FP and FN rates) and allows the implementation in production environments. Our results, which are the first real-time detection approach to be presented in the literature, highlight the feasibility of the proposed architecture that achieves accuracy rates similar to those of traditional offline techniques.
- We present a novel CNN-based detection architecture (PPCensor) suitable for mobile (resource-constrained) end-user devices. We implemented our prototype as a video streaming proxy, so that the CNN model is executed remotely. It does not require device modifications or additional processing on the device, and its working is transparent to the end-user.

The remainder of this paper is organized into the following sections. Section 2 presents work related to CNN-based detection techniques and object detection approaches. Section 3 evaluates current state-of-the-art pornographic video-detection techniques. Section 4 presents PPCensor, which is subsequently evaluated in Section 5. Finally, Section 6 summarizes our main conclusions.

2. Related work

Most video classification techniques split a video into several frames [10], and each frame is classified individually, applying an appropriate technique. These classifications are typically performed by three sequential modules: frame extractor, frame classification, and decision. The frame extractor module extracts the video frames to be classified. The frame classification module classifies each extracted video frame individually as normal or pornographic. Finally, the decision module classifies the video based on the individual frame classifications; in general, the video is classified according to the majority of the frame classifications.

2.1. Convolutional neural networks

In recent years, several detection techniques have been proposed for the frame classification task. In this context, CNN-based detection approaches have become cutting-edge procedures for image processing [13]. A CNN typically consists of three types of layers: convolutional, pooling, and fully connected [17]; its architecture varies in terms of organization and number of layers used.

The most important CNNs for image processing include the ResNet [18] and Inception [19] architectures. In both cases, it is necessary to initialize the configuration parameters, such as the number of epochs, learning rate, decay, and momentum [13]. To improve the detection accuracy and reduce the training time, several researchers have relied on pre-trained CNN architectures through a process known as transfer learning [20,21]. For images, CNNs use the training weights obtained from the ImageNet dataset for transfer learning, a dataset with more than 1.2 million images divided into 1000 classes [22].

The classification procedure begins with an input layer, which receives the raw image, i.e., a matrix of pixels, as input. The resulting data are passed to each subsequent layer until the output is reached. Finally, the output layer generates a probability vector for each classification label, i.e., classification probabilities are produced for normal or pornographic frames.

In the literature, a common approach to improving the accuracy of CNNs is to increase their network complexity. Several studies have considered increasing the number of layers, thereby increasing the demand for processing and memory size. It is common to rely on dedicated hardware equipped with the latest graphics processing units (GPUs). However, such approaches pose a challenge to resource-constrained devices.

Over the last several years, the detection of pornographic content in video frames has been widely studied. However, real-time detection is still in its infancy, especially for approaches intended for deployment on end-user devices. Most of the proposed approaches are based on skin tone or CNN [23].

A lightweight skin-tone-based approach, NuDetective [24], was proposed by Brazilian Federal Police experts. Their software uses the Ap-Apid algorithm [25] to automatically detect nudity in images according to the identified skin tones. NuDetective was developed as a forensic tool to help recover pornographic

files from seized computers. In contrast, Nian et al. [26] detected pornographic images using a CNN architecture trained using transfer learning [20,21] and a training dataset containing 48,600 images (13,300 pornographic and 35,300 normal) collected from the internet. The authors employed a medium-level representation and an adjustment of training data as their two main strategies. The main contribution of Nian's proposal is the application of the "fixed point algorithm" [27], which allows the detection of different image sizes.

The detection of pornographic content, in general, has been evaluated using publicly available datasets such as Pornography-2k [28], Sensitive [9], and NPDI [29]. Moustafa et al. [5] applied several CNN architectures, including AlexNet [22] and GoogleNet [30], for the detection of pornographic videos. In their proposal, videos were categorized as pornographic if the majority of their frames were classified as those containing inappropriate content, as evaluated based on the NPDI [29] dataset.

Another approach was proposed by Yahoo, namely, Yahoo Detector [6], which is a publicly available tool for finding not-safe-for-work images. Their technique applies a ResNet50 [18] CNN architecture trained using transfer learning [20,21], based on ImageNet [22]. Although this technique yields reasonably accurate results, the used dataset was not shared, and therefore, cannot be compared with other approaches. In addition, Vitorino et al. [31] used a CNN architecture (GoogleNet [30]) trained using transfer learning and applied it to the publicly available Pornography-2k dataset [28]. Using this dataset, previously proposed techniques can be evaluated for accuracy comparisons. As an example, Yahoo Detector achieves an accuracy of only 88% when this dataset is used [31].

Perez et al. [7] evaluated their technique using the Pornography-2k [28] and NPDI [29] datasets. The authors proposed the use of static and motion-based features for CNN-based video classification. Their approach significantly improved the detection accuracy, yielding a classification accuracy of 97.9% for the NPDI [29] dataset. However, their approach cannot be implemented in real time because it requires all video frames for the extraction of motion-based features.

Wehrmann et al. [8] proposed a deep-learning architecture, called ACORDE, consisting of a CNN and recurring networks for pornographic content detection in video frames. Their experiments were performed using the NPDI [29] dataset and yielded reasonably accurate results, reducing the number of FPs and FNs of other approaches that utilized the same dataset.

In recent years, some authors have attempted to detect pornographic content in video frames in real time. For instance, Li et al. [32] proposed a system for the real-time detection of abnormal behaviors in live broadcasting platforms. The authors found abnormal rooms through several indicators, such as the degree of a scenery change, real-time comments, and threshold-based abnormal traffic detection. The authors were able to automatically identify abnormal rooms, but their technique ultimately relies on manual human inspection for detection.

Another approach was proposed by Wang et al. [33], which relies on multimodal features, such as video, motion, and sound, for real-time detection of pornographic content. In their study, the authors were able to provide a reasonable level of accuracy and a high detection throughput in broadcasting platforms, with a low FP rate. However, the authors relied on dedicated hardware for evaluation and thus did not address the execution of their technique on resource-constrained devices. Similarly, in a study conducted by Singh et al. [34], video scenes, i.e., sequences of frames, were detected, leveraging a long short-term memory autoencoder fed with a CNN video representation. Although the authors' proposal does not demand the entire video sequence for detection purposes, the deployment of their technique on resource-constrained devices was not addressed.

Although some proposals have enabled the real-time detection of pornographic content, the impacts of FPs on the user experience, as well as the deployment of such techniques on resource-constrained devices, remain an unaddressed challenge.

2.2. Object detection using CNN-based techniques

Traditional CNN-based video classification techniques classify a given input image as either belonging to or not belonging to a particular class (e.g., pornographic or normal). In specific applications, a given image may contain several regions of interest. An example application is a human detection system, in which a single input image may include several persons for identification. In such a case, the goal is the identification of several image regions as opposed to the classification of the entire image.

CNN-based approaches have also been used for object detection tasks in images [35]. The objective is to identify a bounding-box region (image region) that correctly includes the occurrence of the object in the image. Several approaches have been proposed to achieve such a goal, including faster R-CNN [36], single-shot detector (SSD) [37], and You Only Look Once detection [38]. This objective differs from the goal of feature extractors that handles different sizes of input test images [39].

Similar to classification techniques, object detectors, such as faster R-CNNs [36] are also training-based. In Faster R-CNN, the features are first extracted from images and sent to a region proposal network (RPN). Second, the detection is performed for each location, computing the probability of the existence of the searched object. In contrast, the SSD architecture is executed in a single stage, considered fast and appropriate for resource-constrained hardware [39] such as smartphones and tablets.

To improve the detection accuracy, researchers frequently rely on a complex CNN architecture such as Faster R-CNN. As with image-based classification, it is possible to train object detectors with transfer learning [20,21], which facilitates weight refinement using pre-trained CNNs. In general, transfer learning in object detection is achieved using the COCO dataset [40], which consists of 330,000 images divided into 80 classes (objects).

In general, CNN-based object detectors are evaluated for their detection throughput and mean average precision (mAP). The mAP is computed according to the intersection over the union (IoU). In the literature, in general, it is applied at a threshold of 0.5 for each object bounding box. The IoU establishes the region of the identified bounding box that intersects with the target object region [35].

In recent years, several CNN architectures have been proposed for object detection tasks and utilized in several different fields. For instance, used for the recognition of vehicles [41–43], people [44], water surface objects [45], and have even been used for crime scene analysis [46].

Braun et al. [47] applied state-of-the-art object detection CNNs, including Faster R-CNN and SSD, for human recognition. In their study, the authors were able to reach a reasonable level of accuracy when detecting a person in an image. In contrast, for the identification of individual parts, Yang et al. [48] applied an SSD architecture for hand identification, reaching a reasonable accuracy rate. Another approach was proposed by Guo et al. [49] for the identification of facial regions in images through an object-detection CNN architecture.

The applicability of CNN-based object detection techniques for the identification of private parts for pornographic detection remains an open challenge. For object detection in pornography, Ou et al. [9] attempted to classify images/videos by utilizing a deep multi-context network (DMCNet) [50], which is a hierarchical method that applies the feature output of a CNN model to train the RPN in Faster R-CNN object detection. For the purpose

of comparison, the authors generated a Sensitive dataset [9] and compared it with other datasets. Although the authors applied an object-detection CNN architecture, they did not detect any private parts.

Several approaches have been proposed for the detection of pornographic content in video frames. However, their implementation in real-world applications remains an open issue because to increase the level of accuracy, researchers typically extract features over several frames [5,6] or significantly increase the CNN complexity [7–9]. Consequently, real-time detection is still very limited in the literature.

A pornography detection approach based on object detection techniques can significantly improve the user experience because an FP only causes an area of the video to be obfuscated instead of the entire set of video frames being restricted. There are only a few studies that have used object detection to locate private parts or to detect pornographic images.

Our proposal, PPCensor, presents two main benefits when compared to other state-of-the-art techniques. First, unlike traditional pornographic content detection, this is the first work to tackle pornographic detection as an object detection problem. As our contribution to this field, private human parts can be individually identified and properly obfuscated inside a video frame. Consequently, when an FP occurs, only a small section of the video frame is obfuscated; the entire video is not blocked, as in the case of current techniques. Second, unlike related work, we consider the resource-restricted nature of the end user's device. PPCensor runs as a video streaming proxy and is applied transparently, not generating additional processing for the end user's device.

3. Problem statement

In this section, we evaluate the classification performance of publicly available detection approaches for pornographic content. More specifically, we first describe the used pornographic datasets and then evaluate the accuracy and detection throughput of the NuDetective [24] and Yahoo Detector [6] techniques.

3.1. Datasets

The methods used for detecting pornography are typically evaluated against a single dataset. Thus, the accuracy obtained is not demonstrated in production (real-world) environments. In addition, the use of a single dataset can introduce overfitting in detection techniques because a single dataset is prone to several different types of specificities such as the environment in which it was acquired, the extracted quality and encoding, the video length, and other media characteristics.

We evaluated the detection of pornographic content in video frames using two datasets. Specifically, we considered the Pornography-2k [28] and UCF101 [51] datasets. The prior dataset is widely used as a reference for pornographic detection schemes and consists of 2000 videos, of which 1000 are related to pornography, and the other 1000 are normal videos. In contrast, the latter dataset consists of 13,320 normal videos. The objective is to obtain a realistic detection reference for the FP and FN rates. The FP rate indicates the proportion of normal video frames incorrectly classified as pornographic, whereas the FN rate indicates the proportion of pornographic video frames incorrectly classified as normal. Both datasets include individuals of different ethnicities involved in different activities and with varying levels of video quality.

The video frames of the dataset were extracted at an interval of 10 frames, i.e., a frame was extracted for further analysis approximately every 0.3 s. In total, the Pornography-2k dataset consisted of 1,376,041 frames, of which 389,808 frames are normal, and 986,233 frames are pornographic, whereas the UCF101 [51] dataset consisted of videos with 240,810 normal frames.

Table 1
Accuracy and throughput of pornography detection approaches.

Detection technique	Detection throughput (Frames per second)	Dataset		
		Pornography-2k		UCF101
		FP (%)	FN (%)	FP (%)
Yahoo Detector [6]	3.67	12.80	5.53	1.61
NuDetective [24]	411.64	58.48	3.06	23.25

3.2. Reliability of pornography detection schemes

Two pornography detection approaches, Yahoo Detector [6] and NuDetective [24] were evaluated using the built datasets. The evaluated techniques were executed on a desktop computer equipped with an Intel i7 CPU (quad-core), 16 GB of memory, and an Nvidia Titan-XP GPU.

Table 1 presents the throughput and detection accuracy results for Yahoo Detector and NuDetective. It can be seen that NuDetective significantly outperforms Yahoo Detector in terms of throughput, reaching up to 411 FPS, whereas Yahoo Detector reaches only 3.67 FPS. However, Yahoo Detector has a significantly improved detection accuracy, reaching 45.68% and 21.64% smaller FP rates for Pornography-2k and UCF101, respectively, when compared to NuDetective. In contrast, for the FN rate, NuDetective outperforms Yahoo Detector by 2.47% because NuDetective performs detection based on skin tone. Consequently, the detection throughput is significantly improved, but the FP rates are also considerably higher than that of a CNN-based approach.

3.3. Discussion

The evaluation of current pornographic detection techniques for videos shows that they cannot handle real-time detection even when performed using dedicated hardware. These techniques either have a low throughput or significantly high error rates. Running them on resource-constrained devices, environments in which pornographic content is often shared and watched, is not feasible owing to low accuracy or high computational demand. If they have a high FP rate, the user will no longer trust the detection mechanism—recall that the best FP rate was obtained using Yahoo Detector (Table 1). In addition, as these techniques classify an entire video as pornographic or normal, an FP causes the user not to be able to watch the entire video.

4. Private parts censor

To address the real-time detection requirement and the lack of reliability in the classification of pornographic content in videos, we present the Private Parts Censor (PPCensor), whose objective is real-time detection and ease of use without the need for additional processing on the end-user's device. PPCensor runs on dedicated and general-purpose hardware, and the process of detection is transparent to the end-user. PPCensor's operation proceeds in two main stages: video-streaming query and handling and pornographic object detection.

The first stage addresses device query parsing, video download, video filtering, and related filtered responses. The process is entirely executed on a proxy server. Therefore, the only requirement is that the end-user configure the device settings to redirect network queries to the PPCensor server. The detection process does not require additional device processing and can be readily executed even on resource-constrained devices. For instance, a user responsible for an underaged user can configure the device that they want to protect.

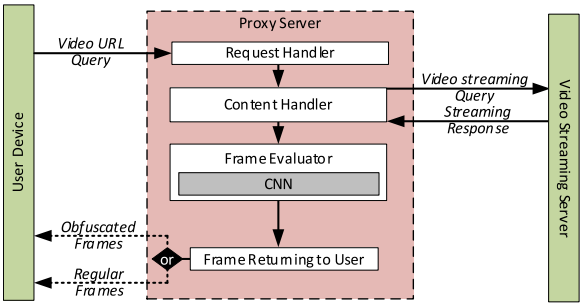


Fig. 1. PPCensor processing architecture.

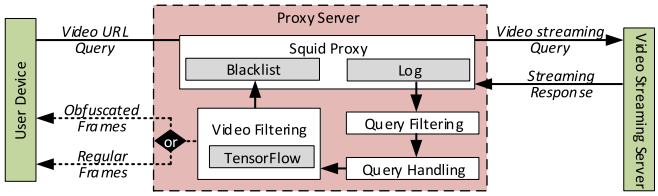


Fig. 2. PPCensor prototype implementation architecture.

The second stage is responsible for presenting a user-friendly obfuscation approach. PPCensor is the first approach that addresses the task of pornography detection as an object detection problem. It is based on the insight that a private part (explicit nudity) can be identified as an object and properly filtered (obfuscated) inside a video frame. As a result, even if a misclassification occurs, the user experience is not significantly degraded because PPCensor deals with inappropriate content by obfuscating (blurring) private parts instead of blocking the entire video.

PPCensor processing architecture is illustrated in Fig. 1. These two stages, including the architectural components, object detection techniques applied to the detection of pornographic video frames and the implementation of the proposed prototype, are detailed in the following subsections.

4.1. Pornographic object detection

Before examining the details of PPCensor's processing architecture, we describe its novel detection approach. Unlike related works that address the detection of pornographic videos through the analysis of entire video frames, PPCensor addresses the detection of pornographic content as an object detection problem. Therefore, PPCensor provides two essential benefits: improved detection throughput and easy-to-use detection. First, the detection of image objects can be achieved in near real time because several CNN architectures are used to handle object detection tasks as classification problems, significantly increasing the detection throughput. Second, filtering specific regions of the image (objects) instead of the entire image substantially improves the user experience because if an FP occurs, only part of the video image is obfuscated, instead of the entire video being blocked.

PPCensor considers each private part, either of a female or male, as an object for detection. Thus, the objective of PPCensor is to recognize four classes of specific objects considered Private Parts: male and female sexual organs (penises and vaginas), female breasts, and buttocks. It is important to note that these private parts are not shown in normal video streams, but they are often featured in pornographic videos. For example, in a single frame, an explicitly pornographic video can show all these private parts.

When considering private parts as objects for detection, PPCensor can detect pornographic videos using object detection techniques based on CNN. Videos can come from any URL on the internet.

4.2. Processing architecture

The objective of using PPCensor is to perform object detection on pornographic content in real time without any additional processing on the user's device. Therefore, PPCensor is implemented as a proxy server, whose processing architecture is illustrated in Fig. 1.

The architecture considers a set of user devices for which video media needs to be evaluated and filtered. For this purpose, the user must only configure their device to redirect the queries to the PPCensor proxy server using it as a web proxy. Therefore, all streaming video queries made by the device are sent to PPCensor. The PPCensor server, in turn, receives the queries through the request handler module, responsible for receiving and analyzing URL (Uniform Resource Locator) queries. As such, the purpose of the module is to establish a video query URL for further processing.

Subsequently, the content handler module queries the video streaming server for the related video frames. It downloads the streaming frames to the proxy server and forwards it to the next module, the content evaluator. This module, in turn, performs the proposed pornographic object detection and appropriately labels all identified image objects (described in Section 4.1). Finally, the content filter module filters the frames in real time by, for example, blurring the identified private parts within each frame.

4.3. Prototype

A PPCensor prototype was implemented and deployed in a multithreaded process, as shown in Fig. 2. The prototype considers three main entities: the end-user device, a proxy server (an implementation of PPCensor), and a video streaming server.

The video URL query process is responsible for the video query in PPCensor. In a production deployment configuration, a video URL query is represented by a request from the end user's device. To that end, a Python program was implemented to perform various video URL queries. All conducted queries are forwarded to the PPCensor (proxy) server implemented using Squid Proxy [52], a well-known network proxy server. The video URL lookup node is configured to use PPCensor as its proxy server.

The PPCensor server is implemented using three main processes. First, the proxy is implemented using Squid Proxy [52] version 4.5. This tool is available to the public and is widely used as a web proxy on Linux-based servers. Over time, Squid Proxy logs all queries in a log file, which is analyzed and interpreted by another process, i.e., the query filtering and handling module. These processes are Python modules and continuously filter the Squid Proxy log. When a video URL is found, the prototype checks the URL to determine if it was already verified, using a cache-based blacklist and whitelist. If the queried URL was not verified, the corresponding media URL was downloaded from the video streaming server through the Pafy API [53], version 0.5.1. The process downloads a video at 0.5 FPS. A single frame is downloaded for each 2-second video interval, which is temporarily cached in a folder for further processing. Finally, the proposed pornographic object detection approach is implemented using Python in another process. This process runs the TensorFlow object detection API [54] using TensorFlow-GPU version 1.14. Thus, the module is executed continuously. When a new video frame is downloaded, the CNN model is applied, and the identified pornographic objects are labeled. The module also allows blacklisting of videos via

the Squid Proxy blacklisting feature. Therefore, a pornography-related URL can also be blocked in real time, if required by parental control software, for example.

The video streaming server is responsible for answering both video URL queries and PPCensor queries. It is important to note that if the PPCensor server blacklist finds a pornography-related URL, the download from the blacklisted URL can be promptly interrupted by parental control. In this case, the end-user will no longer be able to access the video, even if the video stream is currently being downloaded. PPCensor, by default, blurs (obfuscates) only the private parts (objects) within a video frame.

5. Evaluation

We present the PPCensor analysis by answering four research questions (RQ). (RQ1) Is it possible to treat the detection of a pornographic video as an object detection task? (RQ2) How challenging is the detection of each private part? (RQ3) How does PPCensor perform image classification tasks compared to state-of-the-art CNN? (RQ4) Does PPCensor facilitate real-time detection of pornographic video streams?

The following subsections outline the details of the dataset developed in our study. Additionally, the accuracy and throughput of PPCensor are evaluated.

5.1. Private parts object dataset

In general, the data provided in the literature is labeled for the evaluation of schemes for detecting pornographic images or videos, rather than objects. Consequently, to evaluate the proposed PPCensor detection mechanism, we built a new dataset, the PPO (Private Parts Object Dataset).

The PPO dataset was based on the well-known Pornography-2k dataset [28]. Since we treat pornography detection as an object detection problem, we preprocessed the 1000 pornographic videos in Pornography-2k. The preprocessing task filters the video frames according to their content. Thus, only explicitly pornographic images are selected. For this subset, we manually select the regions in each video frame that show a penis, a vagina, female breasts, or buttocks. To achieve this goal, we define a bounding box region that contains an object that is considered pornographic.

The final dataset consists of 50,870 pornographic frames, within which we selected 52,215 objects containing 13,607 penises, 11,346 vaginas, 13,963 female breasts, and 13,299 buttocks. Note that a single video frame may contain more than one target object. For the model building procedure, the dataset was divided into three parts: training, validation, and testing; the parts of the dataset were made using 60%, 20%, and 20% of the originally built dataset, respectively. Consequently, each part of the dataset is composed of mutually exclusive videos, each with their pornographic objects, thus allowing an adequate evaluation of the method. The training dataset is used for training purposes, the validation dataset is applied to the model training adjustment process, and the test dataset is used for the final evaluation of the model.

5.2. Accuracy of PPCensor

To evaluate the feasibility of PPCensor object-based detection, we first address RQ1, using the PPO dataset and several CNN-based object detection approaches. The evaluations were performed on a desktop computer equipped with an Intel i7 CPU (quad-core), 16 GB of memory, and an Nvidia Titan-XP GPU.

All evaluated CNN architectures were trained using transfer learning [20,21]. CNN used the weights obtained from the COCO [40] dataset, which is widely used for object detection purposes. Each CNN architecture was executed in 400,000 steps. Learning and decay rates were defined empirically for each architecture after several evaluations. The CNN-based techniques were evaluated according to their detection throughput and mAP, as computed according to an IoU threshold of 0.5 to each object bounding box.

Table 2
Evaluation of PPCensor performance in pornographic object detection.

CNN architecture	Detection throughput (Frames per second)	mAP (IoU 0.5)
Faster R-CNN Inception v2	10.87	63.50
Faster R-CNN NAS	1.19	56.53
Faster R-CNN ResNet 50	8.33	62.55
SSD MobileNet	59.55	55.92

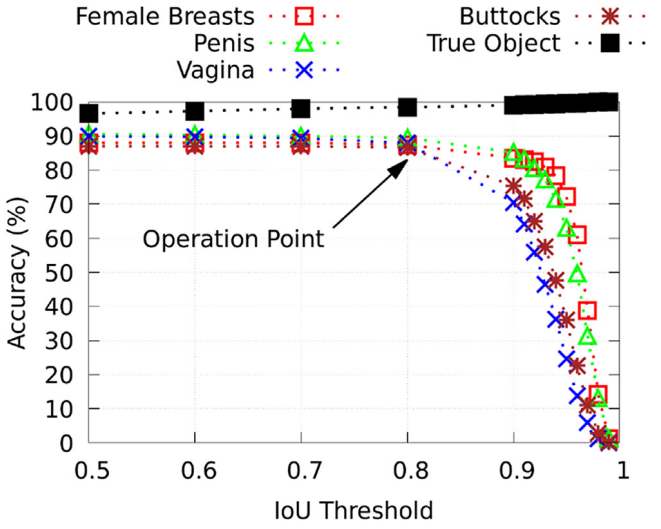


Fig. 3. Faster R-CNN Inception IoU and trade-off of inaccuracy.

Significant differences between the accuracy (mAP) values were obtained for each evaluated CNN architecture, as shown in Table 2. It is important to note that the mAP considers an entire intersection of identified objects of at least 50% in this evaluation. However, in a production environment, the identified bounding box region can be expanded to improve the accuracy of the resulting model without a significant change in the user experience. The Faster R-CNN Inception's mAP architecture shows that it is possible to deal with streaming pornographic video as an object detection task.

In Fig. 3, we investigate the relationship between the IoU threshold and the accuracy rates. It is possible to observe that varying the IoU threshold can improve the accuracy of object detection up to an IoU threshold of 0.8. However, increasing its value further does not significantly improve the true object ratio, while it does significantly decrease the object detection accuracy. Using a 0.8 IoU threshold value improves the object detection for up 3.84% when compared to a 0.5 IoU threshold.

To answer RQ2, we inspect the detection rate of each private part. We inspect the confusion matrix for the Faster R-CNN Inception architecture with an IoU of 0.8 (Operation Point, Fig. 3), as shown in Table 3. It can be seen that private parts are generally not misclassified; about 11% of cases (on average) show as FNs. This means that CNN is able to identify which type of private part was found, although three objects have about 1.5% of false positives (on average): female breasts, vaginas, and penises.

Fig. 4 shows examples of pornographic images labeled using the Faster R-CNN Inception architecture, in which it is possible

Fig. 4. Examples of private parts after identification by PPCensor with Faster R-CNN Inception (IoU 0.8).

Table 3
Confusion matrix for Faster R-CNN Inception architecture (IoU 0.8).

Object class	Classified as				*False Negative (FN)*
	Female breasts	*Penis*	*Vagina*	*Buttocks*	
Female breasts	2186	38	0	5	271
Penis	2	2236	11	4	247
Vagina	0	4	2199	39	258
Buttocks	5	7	38	2165	285

to observe how the proposed object detection approach positively affects the user experience. The regions in the images with private parts are appropriately identified and obfuscated, and the user experience is not significantly degraded, because the user is still able to identify the context of the scene without being exposed to explicit nudity.

Fig. 5 shows examples of non-pornographic images with misrecognized objects (FPs) when the Faster R-CNN Inception architecture is applied. Common examples of FPs include an arm or a finger misidentified as a penis (Fig. 5-c, d, and h) and rounded regions being incorrectly labeled as female breasts (Fig. 5-a, e, and g). In Fig. 5, it can be seen that the incorrect identification of an object does not significantly degrade the end-user experience because only a small part of the video frame is obfuscated. We pixelated people's faces in order to preserve their identification in Figs. 4-e, 5-a, b, c, and d.

Under certain circumstances, the system parental control administrator may prefer to block the entire video, or parts of it, rather than obfuscating specific regions of the video frame (object) identified by PPCensor. For example, in the case of explicitly pornographic videos, it is preferable to block access to the entire video, instead of allowing access to filtered (obfuscated) video sequences. Therefore, to address RQ3, we also evaluated our proposed approach by applying it to image classification. PPCensor was customized to classify a video frame as pornographic if any private part was identified within the video frame. The PPCensor results were compared with the state-of-the-art CNN architectures used for image classification tasks.

The Pornography-2k dataset was divided into training, validation, and testing datasets, comprising 60%, 20%, and 20% of the original dataset, respectively. The CNNs were trained with transfer learning [20,21]. In addition, a weight adjustment was applied using the validation dataset; finally, the final accuracy was measured using the testing dataset. The CNN architectures (CaffeNet [55], AlexNet [22], and Inception [19]) were executed for 400,000 steps. The learning and decay rates were empirically defined individually for each architecture after several evaluations. The obtained models were also evaluated using the UCF101 dataset; in this case, a pornography — classified video frame is considered an FP since the dataset consisted of only normal video frames.

Table 4 shows the PPCensor performance compared to traditional image detection techniques for the Pornography-2k and UCF101 datasets. Surprisingly, PPCensor outperforms all the evaluated state-of-the-art techniques for the Pornography-2k dataset in terms of FP, while also outperforming all other approaches except the Yahoo Detector for the UCF101 dataset.

PPCensor also shows similar results to other techniques regarding its FN rates, reaching 2.34% for the Pornography-2k dataset, an increase of only 0.31% compared with the best CNN architecture. However, the detection throughput of PPCensor is significantly worse than the other CNN architectures. It can be noticed that the detection performance of our proposed technique can be improved by applying other CNN architectures to detect objects (as shown in Table 2).

The evaluation of video frames is only part of the PPCensor architecture (Fig. 2). Therefore, despite the use of CNN with high detection throughput, the rest of the pipeline should also be improved. Specifically, real-time video download and filtering should be addressed. Therefore, in the next subsection, the overall throughput of the PPCensor architecture and the scalability of the connection for real-time video classification tasks are evaluated.

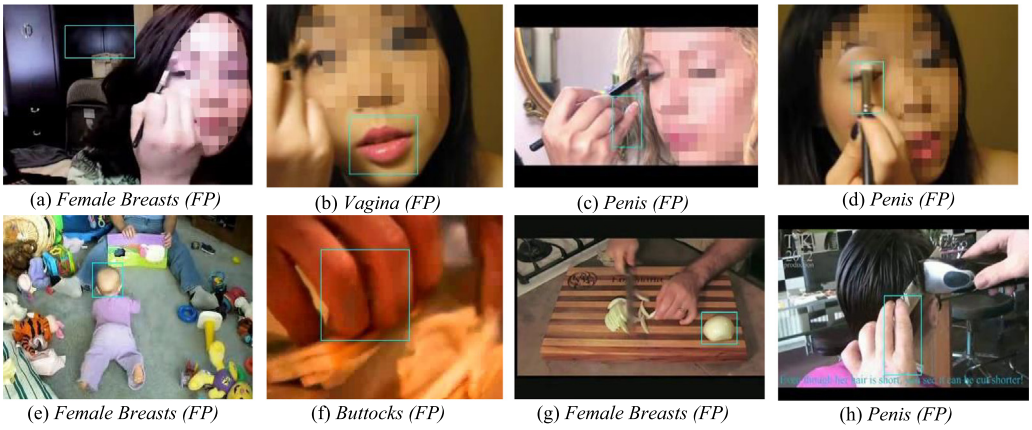
5.3. PPCensor performance and scalability

We evaluated whether PPCensor can perform transparent and real-time detection of pornographic content using a proxy server, without any additional processing on the end-user device, to answer RQ4. The PPCensor node was equipped with an Nvidia Titan-XP GPU, which ran the Faster R-CNN Inception architecture with an mAP threshold of 0.8 (for implementation details, refer to Section 4.3).

Fig. 6 shows the trade-off between the video length and the video download time as a function of the number of connections. It can be observed that PPCensor can download a video for up to ten user connections in real time, as the download time is always shorter than the video length. In addition, the download

Table 4
Comparison of PPCensor and CNN approaches when applied to image classification tasks.

Detection technique	Detection throughput (Frames per second)	Dataset		
		Pornography-2k (Test dataset)		UCF101
		FP (%)	FN (%)	FP (%)
Yahoo Detector [6]	3.67	12.80	5.53	1.61
NuDetective [24]	411.64	58.48	3.06	23.25
CaffeNet [55]	63.97	3.41	3.70	9.68
AlexNet [22]	61.12	5.08	4.61	13.69
Inception [19]	18.29	2.46	2.03	8.03
PPCensor (Faster R-CNN Inception)	10.87	1.66	2.34	3.41



(a) *Female Breasts (FP)* (b) *Vagina (FP)* (c) *Penis (FP)* (d) *Penis (FP)*

(e) *Female Breasts (FP)* (f) *Buttocks (FP)* (g) *Female Breasts (FP)* (h) *Penis (FP)*

Fig. 5. Common FP examples from UCF101 dataset when using Faster R-CNN Inception (IoU 0.8).

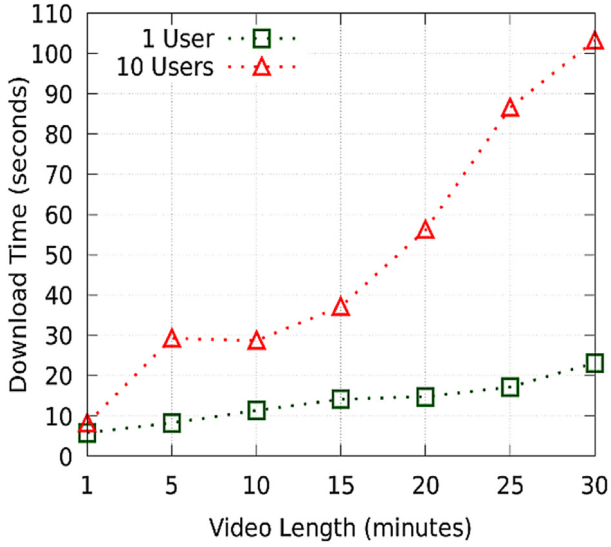


Fig. 6. PPCensor video download time.

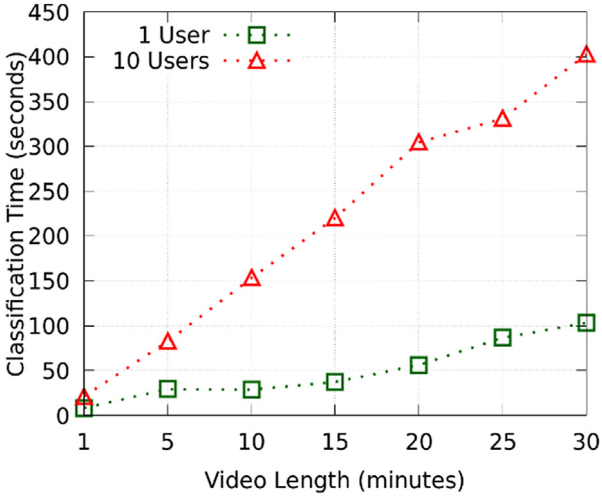


Fig. 7. PPCensor video classification time.

time increases linearly for a single-user scenario. At the same time, there is a significant increase in execution time for ten users because the video download is a network- and CPU-bound. Since the evaluated node contains only eight threads, the execution time decreases significantly below a specific video length limit (20 min).

Fig. 7 shows the trade-off between video length and classification time. It can be observed that the execution time increases almost linearly for both single-user and ten-user scenarios. This behavior is mainly because the classification task is GPU-bound. The evaluation of video frames is performed individually, rather than simultaneously, as only one model is loaded in memory in the prototype implementation.

Fig. 8 shows the trade-off between video time and processing time for the entire PPCensor process, including the video download and frame classification. Detection can be performed in real-time for single- and ten-user scenarios because the process execution time remains shorter than the video length. In addition, the required processing time does not increase significantly compared to the classification time (Fig. 7). This characteristic is due to the CPU-bound nature of the download process and the GPU-bound nature of the classification task. Therefore, the processes of download and classification do not hinder each other. As a result, PPCensor allows the detection of the pornographic video in near

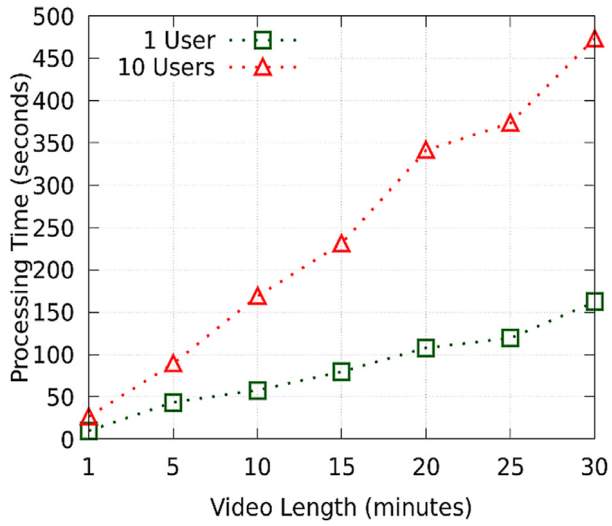


Fig. 8. PPCensor processing time.

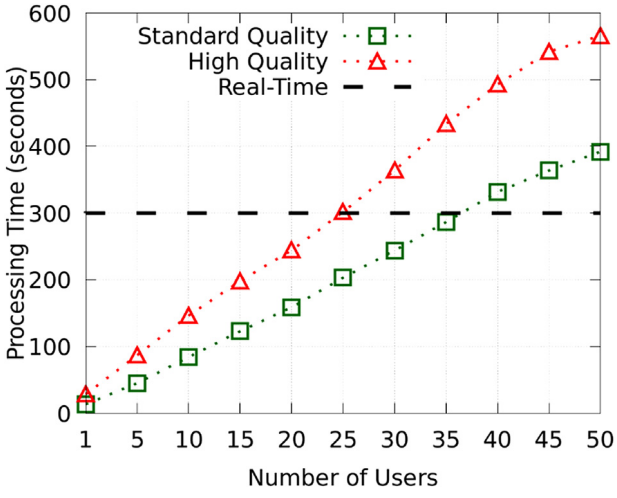


Fig. 9. PPCensor processing time for 5-min-long standard- and high-quality videos. The proposed approach can handle, in real time, 35 users viewing standard-quality video sequences or 25 users viewing high-quality video sequences.

real time, without any additional processing on the end user's device.

Finally, Fig. 9 shows the relationship between the number of users and the process execution time for 5-min streaming video scenarios. In these cases, PPCensor can accommodate, in near real time, up to 35 users watching a video in standard quality or up to 25 users watching a video in high quality. Thus, PPCensor transparently detects pornographic video streams in real time for up to 35 users without incurring any additional device processing demands. At the same time, the detection is performed in a user-friendly manner. It is important to note that the prototype implementation can be adjusted to improve observed issues, further increasing the detection throughput.

6. Conclusion

Current publicly available tools for detecting pornographic content, when streaming videos do not enable transparent detection, are not easy to deploy in real-world environments. In this proposal, we present PPCensor, which is based on two main insights: detection and obfuscation of pornographic content. First, PPCensor treats the identification of pornographic content as an object detection problem. Such an approach provides user-friendly detection without significantly impacting the video watching experience even when an FP occurs or when scenes are showing private parts. An evaluation based on an analysis of more than 50 thousand video frames of (manually labeled) private parts revealed that the proposed technique could detect pornographic content in near real time. In addition, the technique generated similar results to those obtained from state-of-the-art proposals used for image classification tasks. Second, the PPCensor architecture is implemented as a video streaming proxy server; therefore, it does not incur additional processing on the end user's device, while performing transparent private part detection. An immediate positive impact of this approach is its usability on mobile devices such as smartphones and tablets. Our proposal is the first implementation that detects private parts as objects in real time for resource-constrained end-user devices. The evaluation, which was carried out on a high-end desktop computer with a proxy server for video streaming, shows that PPCensor can accommodate up to 35 simultaneous connections from end-users in near real time. The PPO dataset and PPCensor source code are publicly available for download at https://secplab.ppgia.pucpr.br/ppcensor.

CRediT authorship contribution statement

Jackson Mallmann: Conceptualization, Investigation, Methodology, Writing - original draft. **Altair Olivo Santin:** Conceptualization, Investigation, Writing - review & editing. **Eduardo Kugler Viegas:** Data curation, Methodology, Validation, Visualization. **Roger Robson dos Santos:** Software, Visualization. **Jhonatan Geremias:** Data curation, Software, Visualization.

Acknowledgments

The authors thank the Brazilian National Council for Scientific and Technological Development (CNPq) for their partial financial support (grant nº430972/2018-0), NVIDIA Co. for donating the Titan-XP GPU used in the experiments, and URCOP (*Unidade de Repressão aos Crimes de Ódio e à Pornografia Infantil*), a special division of the Federal Police of Brazil for pornography crackdowns, for their support to the project. We especially thank Marcelo da Silva Moreira, Rafaela Vieira Lins Parca, and Cassina Saad de Carvalho from the Federal Police for their extraordinary support of the project's objectives. Jackson Mallmann would like to thank the IFC (Federal Institute Catarinense) and Coordination for the Improvement of Higher Education Personnel (CAPES) for the scholarship, granting nº231/2017.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] S. Statista, How much of the Internet consists of porn? 2019, Available online: https://www.statista.com/chart/16959/share-of-the-internet-that-is-porn/. (Accessed 1 February 2020).
[2] S. Pornhub, 2018 year in review, 2018, Available online: https://www.pornhub.com/insights/2018-year-in-review/. (Accessed 1 February 2020).
[3] P.C.A. America, Prevent child abuse, 2019, Available online: https://preventchildabuse.org/. (Accessed 1 February 2020).
[4] E. Enough, Pornography statistics, 2019, Available online: http://enough.org/stats_porn_industry. (Accessed 1 February 2020).

[5] M. Moustafa, Applying deep learning to classify pornographic images and videos, in: 7th Pacific-Rim Symposium on Image and Video Technology, PSIVT, 2015, Available online: https://www.researchgate.net/conference-event/PSIVT_Image-and-Video-Technology_2015/2645. (Accessed 1 February 2020).

[6] J. Mahadeokar, G. Pesavento, Open sourcing a deep learning solution for detecting nsfw images, 2016, Available online: https://yahooeng.tumblr.com/post/151148689421/open-sourcing-a-deep-learning-solution-for. (Accessed 1 February 2020).

[7] M. Perez, S. Avila, D. Moreira, D. Moraes, V. Testoni, E. Valle, S. Goldenstein, A. Rocha, Video pornography detection through deep learning techniques and motion information, Neurocomputing (ISSN: 0925-2312) 230 (2017) 279–293, http://dx.doi.org/10.1016/j.neucom.2016.12.017.

[8] J. Wehrmann, G.S. Simões, R.C. Barros, V.F. Cavalcante, Adult content detection in videos with convolutional and recurrent neural networks, Neurocomputing 272 (2018) 432–438, http://dx.doi.org/10.1016/j.neucom.2017.07.012.

[9] X. Ou, H. Ling, H. Yu, P. Li, F. Zou, S. Liu, Adult image and video recognition by a deep multicontext network and fine-to-coarse strategy, ACM Trans. Intell. Syst. Technol. 8 (5) (2017) http://dx.doi.org/10.1145/3057733, Article 68.

[10] F. Idris, S. Panchanathan, Review of image and video indexing techniques, J. Vis. Commun. Image Represent. 8 (2) (1997) 146–166, http://dx.doi.org/10.1006/jvci.1997.0355.

[11] Y. Kuroki, T. Nishi, S. Kobayashi, H. Oyaizu, S. Yoshimura, A psychophysical study of improvements in motion-image quality by using high frame rates, J. Soc. Inf. Disp. 15 (2007) http://dx.doi.org/10.1889/1.2451560.

[12] S. Wearsocial, The state of digital in 2019: all the numbers you need to know, 2019, Available online: https://wearesocial.com/blog/2019/04/the-state-of-digital-in-april-2019-all-the-numbers-you-need-to-know. (Accessed 1 February 2020).

[13] W. Rawat, Z. Wang, Deep convolutional neural networks for image classification: A comprehensive review, Neural Comput. 29 (9) (2017) 2352–2449, http://dx.doi.org/10.1162/neco_a_00990.

[14] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M.P. Reyes, M. Shyu, S. Chen, S.S. Iyengar, A survey on deep learning: Algorithms, techniques, and applications, ACM Comput. Surv. 51 (5) (2019) 1–36, http://dx.doi.org/10.1145/3234150.

[15] M. Imani, D. Peroni, Y. Kim, A. Rahimi, T. Rosing, Efficient neural network acceleration on GPGPU using content addressable memory, in: Proceedings of the 2017 Design, Automation and Test in Europe, pp. 1026–1031, http://dx.doi.org/10.23919/DATE.2017.7927141.

[16] H. Park, D. Kim, J. Ahn, S. Yoo, Zero and data reuse-aware fast convolution for deep neural networks on GPU, in: 2016 International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS 2016, 2016, pp. 1–10, http://dx.doi.org/10.1145/2968456.2968476.

[17] A. Karpathy, CS231n: Convolutional neural networks for visual recognition, 2016, Available online: http://cs231n.github.io/convolutional-networks/. (Accessed February 2020).

[18] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Las Vegas, NV, 2016, pp. 770–778, http://dx.doi.org/10.1109/CVPR.2016.90.

[19] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826, https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.308.

[20] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and transferring mid-level image representations using convolutional neural networks, in: Proceedings IEEE Conf. on Computer Vision and Pattern Recognition, 2014, http://dx.doi.org/10.1109/CVPR.2014.222.

[21] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, C. Liu, A survey on deep transfer learning, 2018, http://dx.doi.org/10.1007/978-3-030-01424-7_27.

[22] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90, http://dx.doi.org/10.1145/3065386.

[23] S. Karamizadeh, A. Arabsorkhi, Methods of pornography Detection: Review, in: Proceedings of the 10th International Conference on Computer Modeling and Simulation, ICCMS 2018, ACM, New York, NY, USA, pp. 33–38, http://dx.doi.org/10.1145/3177457.3177484.

[24] M. Polastro, P. Eleuterio, Nudetective: a forensic tool to help combat child pornography through automatic nudity detection, in: Proc. Workshop on Database and Expert Systems Applications, DEXA, 2010, http://dx.doi.org/10.1109/DEXA.2010.74.

[25] R. Ap-apid, An algorithm for nudity detection, in: Proceedings of the 5th Philippine Computing Science Congress, 2005.

[26] F. Nian, T. Li, Y. Wang, M. Xu, J. Wu, Pornographic image detection utilizing deep convolutional neural networks, Neurocomputing 210 (2016) 283–293, http://dx.doi.org/10.1016/j.neucom.2015.09.135.

[27] R. Gordon, A calculated look at fixed-point arithmetic, Embed. Syst. Progr. 11 (4) (1998) 72–79.

[28] D. Moreira, S. Avila, M. Perez, D. Moraes, V. Testoni, E. Valle, S. Goldenstein, A. Rocha, Pornography classification: The hidden clues in video space–time, Forensic Sci. Int. 268 (2016) 46–61, http://dx.doi.org/10.1016/j.forsciint.2016.09.010.

[29] S. Avila, N. Thome, M. Cord, E. Valle, A.A. Araújo, Pooling in image representation: the visual codeword point of view, Comput. Vis. Image Underst. 117 (5) (2013) 453–465, http://dx.doi.org/10.1016/j.cviu.2012.09.007.

[30] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9, http://dx.doi.org/10.1109/CVPR.2015.7298594.

[31] P. Vitorino, M. Avila, S. Perez, A. Rocha, Leveraging deep neural networks to fight child pornography in the age of social media, Proc. J. Vis. Commun. Image Represent. (2018) http://dx.doi.org/10.1016/j.jvcir.2017.12.005.

[32] Y. Li, W. Ren, T. Zhu, Y. Ren, Y. Qin, W. Jie, RIMS: A Real-time and Intelligent Monitoring System for live-broadcasting platforms, Future Gener. Comput. Syst. 87 (2018) 259–266, http://dx.doi.org/10.1016/j.future.2018.04.012.

[33] L. Wang, J. Zhang, Q. Tian, C. Li, L. Zhuo, Porn streamer recognition in live video streaming via attention-gated multimodal deep features, IEEE Trans. Circuits Syst. Video Technol. 1 (2019) http://dx.doi.org/10.1109/TCSVT.2019.2958871.

[34] S. Singh, A.B. Buduru, R. Kaushal, P. Kumaraguru, KidsGUARD: Fine grained approach for child unsafe video representation and detection, in: Proc. ACM Symp. Appl. Comput, 2019, pp. 2104–2111, http://dx.doi.org/10.1145/3297280.3297487, Part F147772.

[35] Z. Zhao, P. Zheng, S. Xu, X. Wu, Object detection with deep learning: A review, IEEE Trans. Neural Netw. Learn. Syst. (2019) 1–21, http://dx.doi.org/10.1109/TNNLS.2018.2876865.

[36] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (6) (2017) 1137–1149, http://dx.doi.org/10.1109/TPAMI.2016.2577031.

[37] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, A.C. Berg, SSD: single shot multibox detector, in: Computer Vision, ECCV 2016, Springer International Publishing, http://dx.doi.org/10.1007/978-3-319-46448-0_2.

[38] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, 2018, CoRR.

[39] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, K. Murphy, Speed/accuracy trade-offs for modern convolutional object detectors, in: The IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2017, http://dx.doi.org/10.1109/CVPR.2017.351.

[40] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft COCO: Common objects in context, in: ECCV, 2014, http://dx.doi.org/10.1007/978-3-319-10602-1_48.

[41] L. Wang, J. Liao, C. Xu, Vehicle detection based on drone images with the improved faster R-CNN, in: Proceedings of the 2019 11th International Conference on Machine Learning and Computing, ICMLC '19, ACM, New York, NY, USA, pp. 466–471, http://dx.doi.org/10.1145/3318299.3318383.

[42] G. Plastiras, C. Kyrkou, T. Theocharides, Efficient ConvNet-based Object detection for unmanned aerial vehicles by selective tile processing, in: Proceedings of the 12th International Conference on Distributed Smart Cameras, ICDSC '18, ACM, New York, NY, USA, http://dx.doi.org/10.1145/3243394.3243692.

[43] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, K. Ouni, Car detection using unmanned aerial vehicles: Comparison between faster R-CNN and YOLOv3, in: Proceeding of 1st International Conference Unmanned Vehicle Syst.-Oman, UVS, 2019, pp. 1–6, http://dx.doi.org/10.1109/UVS.2019.8658300, Feb.

[44] P. Maheshwari, D. Alex, S. Banerjee, S. Behera, S. Panda, Top view person detection and counting for low compute embedded platforms, in: Proceedings of the 2018 the 2nd International Conference on Video and Image Processing, ICVIP 2018, ACM, New York, NY, USA, pp. 35–43, http://dx.doi.org/10.1145/3301506.3301548.

[45] L. Zhang, Y.A. Zhang, Z. Zhang, J. Shen, H. Wang, Real-time water surface object detection based on improved faster R-CNN, 2019, http://dx.doi.org/10.3390/s19163523, https://www.mdpi.com/1424-8220/19/16/3523.

[46] S. Saikia, E. Fidalgo, E. Alegre, L. Fernández-Robles, Object detection for crime scene evidence analysis using deep learning, in: S. Battiato, G. Gallo, R. Schettini, F. Stanco (Eds.), Image Analysis and Processing, ICIAP 2017, Lecture Notes in Computer Science, vol. 10485, Springer, Cham, http://dx.doi.org/10.1007/978-3-319-68548-9_2.

[47] M. Braun, S. Krebs, F. Flohr, D.M. Gavrila, EuroCity persons: A novel benchmark for person detection in traffic scenes, IEEE Trans. Pattern Anal. Mach. Intell. 41 (2019) 1844–1861, http://dx.doi.org/10.1109/TPAMI.2019.2897684.

[48] L. Yang, Z. Qi, Z. Liu, H. Liu, M. Ling, L. Shi, X. Liu, An embedded implementation of CNN-based hand detection and orientation estimation algorithm, Mach. Vis. Appl. 30 (2019) 1071–1082, http://dx.doi.org/10.1007/s00138-019-01038-4.

[49] G. Guo, H. Wang, Y. Yan, J. Zheng, B. Li, A fast face detection method via convolutional neural network, Neurocomputing (2019) http://dx.doi.org/10.1016/j.neucom.2018.02.110.

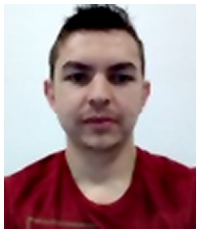
[50] R. Zhao, W. Ouyang, H. Li, X. Wang, Saliency detection by multi-context deep learning, in: Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, CVPR'15, pp. 1265–1274, http://dx.doi.org/10.1109/CVPR.2015.7298731.

[51] K. Soomro, A.R. Zamir, M. Shah, UCF101: A dataset of 101 human actions classes from videos. In the Wild, 2012, CoRR.
[52] Squid development projects, 2020, Available online: http://devel.squid-cache.org/. (Accessed 1 February 2020).
[53] Pafy, 2020, Available online: https://pythonhosted.org/pafy/. (Accessed 1 February 2020).
[54] Tensorflow object detection API, 2020, Available online: https://github.com/tensorflow/models/tree/master/research/object_detection/. (Accessed 1 February 2020).
[55] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: convolutional architecture for fast feature embedding, in: Proceedings of the 22nd ACM International Conference on Multimedia, MM '14, ACM, New York, NY, USA, pp. 675–678, http://dx.doi.org/10.1145/2647868.2654889.

Jackson Mallmann received the BS degree in computer science in 2004 and the MSC degree in computer science in 2011 from PUCPR. He is currently working toward his PhD degree in computer science at PUCPR. His research interests include machine learning, pornography detection and computer security.

Altair Olivo Santin received the BS degree in Computer Engineering from the PUCPR in 1992, the MSc degree from UTFPR in 1996, and the PhD degree from UFSC in 2004. He is a full professor of Graduate Program in Computer Science (PPGIa) and head of Security & Privacy Lab (SecPLab) at PUCPR. He is a member of the IEEE, ACM, and the Brazilian Computer Society.

Eduardo Viegas received the BS degree in computer science in 2013, the MSC degree in computer science in 2016 from PUCPR, and the PhD degree from PUCPR in 2018. He is an associate professor of Graduate Program in Computer Science (PPGIa). His research interests include machine learning, network analytics and computer security.



Roger Robson dos Santos received the BS degree in information systems from PUCPR in 2014 and is currently working toward the MS degree at PUCPR. His research interests include machine learning and computer security.

Jhonatan Geremias received the BS degree in computer science from PUCPR in 2016 and is currently working toward the MS degree at PUCPR. His research interests include machine learning, pornography detection and computer security.