JACKSON MALLMANN

PARTES ÍNTIMAS: DETECÇÃO *ONLINE* E OFUSCAÇÃO PARA *BENCHMARKING* PÚBLICO DE VÍDEO PORNOGRÁFICO

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná, como requisito parcial à obtenção do título de Doutor em Informática.

Orientador: Prof. Dr. Altair Olivo Santin
Coorientador: Prof. Dr. Eduardo Kugler Viegas

CURITIBA

2021

JACKSON MALLMANN

PARTES ÍNTIMAS: DETECÇÃO *ONLINE* E OFUSCAÇÃO PARA *BENCHMARKING* PÚBLICO DE VÍDEO PORNOGRÁFICO

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná, como requisito parcial à obtenção do título de Doutor em Informática.

Área de Concentração: *Ciência da Computação*

Orientador: Prof. Dr. Altair Olivo Santin
Coorientador: Prof. Dr. Eduardo Kugler Viegas

CURITIBA

2021

Reservado para Biblioteca

Ata

A paciência é a melhor maneira de vencer.
Santo Antônio de Pádua

Agradecimentos

A meus Pais, por terem me ensinado o real significado da dignidade.

Minha Esposa, pela compreensão, motivação e auxílio para realização do curso de Doutorado.

Meus orientadores, que em nenhum momento mediram esforços para poderem dar alicerces acadêmicos durante esta jornada.

A todos os Professores do Programa de Pós-Graduação em Informática (PPGIa), pelo excelente trabalho dedicado na construção de meu conhecimento científico. Assim como, a Pontifícia Universidade Católica do Paraná (PUC-PR) por me fornecer a bolsa de pesquisa, o Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro parcial ao projeto, processo: 430972/2018-0, a NVIDIA *Corporation* pela doação de uma GPU Titan-XP usada nos experimentos, a Unidade de Repressão aos Crimes de Ódio e à Pornografia Infantil da Polícia Federal do Brasil (URCOP), ao Instituto Federal Catarinense (IFC) e ao Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela bolsa concedida através do edital n° 231/2017.

A todos os meus amigos. Em especial, aqueles que conheci no laboratório SecPlab. Dentre estes, agradeço ao Prof. Dr. Eduardo Kugler Viegas, por incentivar e proporcionar suporte no entendimento para questões ligadas a estudos realizados ao longo do curso. Não tão menos, Roger Robson dos Santos, por seus incontáveis suportes técnicos.

E a todos que aqui não estão descritos. Aqueles que me incentivaram e, principalmente, respeitaram meus inúmeros momentos de silêncio que este estudo impôs.

Sumário

SUMÁRIO ..... **VII**

LISTA DE FIGURAS ..... **X**

LISTA DE TABELAS ..... **XII**

LISTA DE ABREVIATURAS ..... **XIII**

RESUMO ..... **XV**

ABSTRACT ..... **XVI**

CAPÍTULO 1 INTRODUÇÃO **17**

1.1. Contextualização 17

1.2. Motivação 19

1.3. Hipótese 20

1.4. Objetivos 22

1.5. Contribuições 22

1.6. Publicações 23

1.7. Patente 25

1.8. Organização 25

CAPÍTULO 2 FUNDAMENTAÇÃO **26**

2.1. *Deep Learning* 26

2.2. CNN 27

2.2.1. Detecção de Objetos 30

2.2.2. Parâmetros 32

2.3. GANS 33

2.4. Discussão....34

CAPÍTULO 3 O DESAFIO DA DETECÇÃO *ONLINE* DE VÍDEOS PORNOGRÁFICOS....36

3.1. Bases Utilizadas....36

3.2. A Confiabilidade dos Esquemas de Detecção de Pornografia....38

3.3. Detecção de Objetos para o Ofuscamento....39

3.4. Discussão....42

CAPÍTULO 4 TRABALHOS RELACIONADOS....44

4.1. Aplicações GAN....44

4.2. Detecção de Pornografia....46

4.3. Detector de Objetos....47

4.4. *Pooling*....48

4.5. Discussão....49

CAPÍTULO 5 PARTES ÍNTIMAS: DETECÇÃO *ONLINE* E OFUSCAÇÃO PARA *BENCHMARK* PÚBLICO DE VÍDEO PORNOGRÁFICO....53

5.1. Visão Geral da Proposta....53

5.2. PPCensor: Uma Arquitetura para Detecção *Online* de Pornografia em *Streaming* de Vídeo 55

5.2.1. Detecção de Objetos Pornográficos....56

5.2.2. Arquitetura de Processamento....57

5.3. PPOBench: Publicação de Imagens de Pornografia sem Revelar as Partes Íntimas das Cenas 57

5.3.1. GAN - Reconstituição....58

5.3.2. CNN - Detecção de Imagem Pornográfica....59

5.3.3. Discussão....60

CAPÍTULO 6 AVALIAÇÃO....62

6.1. PPCensor 62
6.1.1. PPO (*Private Parts Object Dataset*) 62
6.1.2. Protótipo 63
6.1.3. Acurácia de PPCensor 65
6.1.4. Performance e Escalabilidade 70
6.1.5. Discussão 73
6.2. PPOBench 73
6.2.1. Bases Formalizadas – Pu e Pt 73
6.2.2. Protótipo 75
6.2.3. Valores de Referência 77
6.2.4. Reconstituição da Base Pt 78
6.2.5. Detecção de Pornografia 80
6.2.6. Discussão 84
CAPÍTULO 7 CONSIDERAÇÕES FINAIS 85
REFERÊNCIAS BIBLIOGRÁFICAS 88
ANEXO I 97

Lista de Figuras

Figura 1. Rede padrão da CNN. Adaptado de [34]. 28
Figura 2. Funcionamento do *Pooling.* 29
Figura 3. Exemplo de rede CNN. Adaptado de [36]. 32
Figura 4. Exemplos de imagens da base Pornography-2k [66]. Na prática, as imagens pornográficas não estão ofuscadas. Elas foram censuradas para efeitos de edição. 37
Figura 5. Exemplos de imagens da base UCF101 [67]. 37
Figura 6. Exemplos de imagens pornográficas da base *baseline*. Na prática, as imagens pornográficas não estão ofuscadas. Elas foram censuradas para efeitos de edição 39
Figura 7. Exemplos de imagens normais da base *baseline.* 40
Figura 8. Exemplos de imagens da base Detector. 40
Figura 9. Exemplos de imagens da base Manual. 40
Figura 10. Método proposto. Adaptado de [21]. 53
Figura 11. Arquitetura de processamento do PPOBench. 58
Figura 12. Exemplo de funcionamento. 60
Figura 13. Exemplos de imagens da base PPO. 63
Figura 14. Arquitetura de implementação do protótipo PPCensor [21]. 64
Figura 15. *Faster R-CNN Inception* IoU e ponto de operação. Adaptado de [21]. 66
Figura 16. *Faster R-CNN Inception* (IoU 0.8) - exemplos de detecção das partes íntimas realizada por PPCensor. Adaptado de [21]. 68
Figura 17. *Faster R-CNN Inception* (IoU 0.8) - exemplos comuns de FP: UCF101. Adaptado de [21]. 69
Figura 18. Tempo para PPCensor fazer o *download* do vídeo. Adaptado de [21]. 71
Figura 19. Tempo para PPCensor fazer a classificação. Adaptado de [21]. 71
Figura 20. Tempo para PPCensor fazer o *download* e classificação do vídeo. Adaptado de [21]. 72
Figura 21. Tempo de processamento do PPCensor para vídeos de 5 minutos e de alta qualidade. Adaptado de [21]. 72
Figura 22. Exemplos de imagens pornográficas (treinamento) – Pu (linha 1) e Pt (linha 2). Na prática, essas imagens não estão ofuscadas. Elas foram censuradas para efeitos de edição. 74

Figura 23. Exemplos de imagens pornográficas (teste) – Pu e Pt. Na prática, essas imagens não estão ofuscadas. Elas foram censuradas para efeitos de edição. 75
Figura 24. Exemplos de imagens não pornográficas (teste) – Pu e Pt. 75
Figura 25. Exemplos de imagens de treinamento da base Pt reconstituída. 78
Figura 26. Exemplos de imagens de teste da base Pt reconstituída. 78
Figura 27. Curva ROC. 82
Figura 28. *Explainable* para imagens teste. 83

Lista de Tabelas

Tabela 1. Resultado – quantidade de parâmetros. Adaptado de [34]. 33
Tabela 2. Acurácia e taxa de transferência das abordagens de detecção de pornografia [21]. 38
Tabela 3. Resultados. 41
Tabela 4. Acurácia da CNN – inserção de imagens pornográficas não ofuscadas. 41
Tabela 5. Comparação entre os trabalhos relacionados. 51
Tabela 6. Avaliação do desempenho do PPCensor na detecção de objetos pornográficos. Adaptado de [21]. 66
Tabela 7. *Faster R-CNN Inception* (IoU 0.8) - matriz de confusão. Adaptado de [21]. ... 67
Tabela 8. Comparação do PPCensor quando aplicado a tarefas de classificação de imagens. Adaptado de [21]. 70
Tabela 9. Resultados: Pu e Pt. 77
Tabela 10. Resultados: reconstituição das imagens. 79
Tabela 11. Resultados. 79
Tabela 12. Resultados: reconstituição das imagens - pix2pixHD [54]. 80
Tabela 13. Inserção/modificação da operação realizada no *Pooling*: Pu. 81
Tabela 14. Inserção/modificação da operação realizada no *Pooling*: Pt reconstituída. pix2pixHD – 250 épocas. 81
Tabela 15. Resumo – Acurácias: Pu e Pt reconstituída. 82

Lista de Abreviaturas

ACC	Acurácia
API	*Application Programming Interface*
CEP	Comitê de Ética e Pesquisa
CGAN	*Conditional* GAN
CNN	*Convolutional Neural Networks*
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CNPq	Conselho Nacional de Desenvolvimento Científico e Tecnológico
Conv	*Convolutions*
CPU	*Central Processing Unit*
DBN	*Deep Belief Networks*
DSFD	*Dual Shot Face Detector*
Fast R-CNN	*Fast Region-based Convolutional Network*
FDF	*Flickr Diverse Faces*
FN	Falso Negativos
FP	Falso Positivos
FA	Função de Ativação
FC	*Full connection*
FCN	*Fully Connected Network*
FNR	Taxa de Falso Negativo
FPR	Taxa de Falso Positivo
GAN	*Generative Adversarial Networks*
GPU	*Graphics Processing Unit*
IA	Inteligência Artificial
IFC	Instituto Federal Catarinense
INPI	Instituto Nacional da Propriedade Industrial
IoU	Interseção sobre a união
ISS	*Internet Security Suites*
LRP	*Layer-wise Relevance Propagation*
mAP	Precisão Média
NB	*Naive Bayes*
NCMEC	*National Center for Missing Exploited Children*
NSFW	*Not-Safe-For-Work*
PPCensor	*Private Parts Censor*
PI	Pornografia Infantil
PPGIa	Programa de Pós-Graduação em Informática
PPO	*Private Parts Object Dataset*
Pt	Base de Imagens de Pornografia Ofuscadas
Pu	Base de Imagens de Pornografia não Ofuscadas
PUC-PR	Pontifícia Universidade Católica do Paraná
ReLU	*Rectified Linear Function*
RPN	*Region Proposal Network*
RCN	*Recurrent Convolution*
RNN	*Recurrent Neural Network*
RvNN	*Recursive Neural Network*
SBSeg	

SSD	Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais *Single Shot Detector*
TPR	Taxa de Verdadeiro Positivo
TNR	Taxa de Verdadeiro Negativo
YOLO	*You Only Look Once*
URCOP	Unidade de Repressão aos Crimes de Ódio e à Pornografia Infantil da Polícia Federal do Brasil
URL	*Uniform Resource Locator*

Resumo

Pesquisas na área de detecção de pornografia são dificultadas pela falta de bases públicas, que poderiam divulgar pornografia; reprimida em alguns países. Neste trabalho é proposto um método para detecção e ofuscação, em tempo real, de partes íntimas em *streaming* de vídeo *online* – PPCensor, e um método para publicação de imagens de pornografia sem revelar as partes íntimas nas cenas, PPOBench. PPCensor aplica a técnica de detecção de objetos usando *Convolutional Neural Network* (CCN) para classificar e censurar, em tempo real, apenas as partes íntimas nas cenas de um vídeo. Usando esta mesma técnica as partes íntimas são ofuscadas e disponibilizadas numa base pública, denominada de *Private Parts Object Dataset* (PPO). PPOBench faz a reconstituição das partes ofuscadas da PPO usando uma *Generative Adversarial Network* (GAN) e uma CCN com o estágio de *Pooling* modificado para permitir o *benchmarking* sem expor publicamente as partes íntimas na base. No protótipo, PPCensor gerou 11,14% menos falsos positivos que o Yahoo Detector, e PPOBench aumentou a acurácia em 15,77% em relação a abordagem com imagens não ofuscadas. Além da contribuição científica de detecção em tempo real, o PPCensor pode ser utilizado no mundo real: em escolas, casas, empresas etc., evitando a exposição das partes íntimas de vídeo pornográfico visualizado *online*, diferentemente da literatura que trabalha *offline* e geralmente censura o vídeo todo. Salienta-se que a contribuição de disponibilização pública de imagens de pornografia com as partes íntimas ofuscadas pode ser utilizada em *benchmarking* por outros pesquisadores da área.

Palavras-chave: detecção e ofuscação de partes íntimas, base pública de pornografia para *benchmark* sem revelar partes íntimas, CNN, GAN, detecção de objetos em vídeo.

Abstract

Research in pornography detection is hardened by the lack of public bases, which could spread pornography; repressed in some countries. This work proposes a method for detecting and blocking, in real time, pornography in online video streaming, namely PPCensor, and a method for publishing pornography images without revealing the intimate parts of the scenes, PPOBench. PPCensor works with object detection technique using Convolutional Neural Network (CCN) to classify and censor, in real time, only the scenes containing the private parts in a video. Using the same technique, private parts are obfuscated and made publicly available, namely Private Parts Object Dataset (PPO). PPOBench reconstitutes the obfuscated parts of the PPO using a Generative Adversarial Network (GAN) and a CCN with the modified Pooling stage to allow benchmarking without publicly exposure of the private parts at the base. In the prototype, PPCensor generated 11.14% less fake positives than Yahoo Detector, and PPOBench increased the accuracy by 15.77% in relation to the approach with dim images. In addition to the scientific contribution of real-time detection, PPCensor can be used in the real world: in schools, homes, companies etc., avoiding the exposure of private parts at pornographic video viewed online, unlike the literature that works offline and generally censors the whole video. It should be noted that the contribution of making pornography images publicly available with the private parts obfuscated can be used in benchmarking by other researchers in this area.

Keywords: detecting and obfuscation the private parts, pornography public base to benchmark without reveal private parts, CNN, GAN, video object detection.

Capítulo 1

Introdução

Este Capítulo apresenta a contextualização do trabalho, motivação, hipótese, objetivos, contribuições e as publicações. Por fim, apresenta-se a organização desse documento.

1.1. Contextualização

Entre os conteúdos disponibilizados na Internet, 30% são classificados como pornográficos [1], sendo que nos últimos anos, a disseminação de vídeos para maiores de idade na Internet, incluindo vídeos com conteúdo explicitamente pornográfico, aumentou significativamente [2]. Por exemplo, em 2018, o *site* pornográfico Pornhub[1], recebeu mais de quatro milhões de *uploads* diários de vídeos [3]. Outrora, vídeos pornográficos podem ser facilmente postados em redes sociais, tais como Facebook[2], Twitter[3], Instagram[4] e YouTube[5], os quais permitem a disponibilização de *Uniform Resource Locators* (URLs) que podem alocar este tipo de conteúdo.

Ainda, verifica-se tendência de usuários consumirem vídeos de forma *online*, principalmente pelo fato do crescimento da Internet e aumento do número de dispositivos com recursos limitados, sendo que os recursos devem atender a mobilidade de usuários, ou seja, em muitos casos, o custo computacional e o consumo de energia são criteriosos. Cita-se como

[1] https://pt.pornhub.com

[2] http://www.facebook.com

[3] http://www.twitter.com

[4] http://www. instagram.com

[5] http://www.youtube.com

exemplo os smartphones e tablets, os quais, conforme pesquisa[6], são maioria em relação a computadores.

A detecção de conteúdo pornográfico em vídeo pode ser realizada aplicando as redes neurais convolucionais (CNN do inglês *Convolutional Neural Network* ou ConvNet) [4], sendo que atualmente, o uso de CNNs encontra-se em várias áreas, tais como de classificação de imagens [5], de detecção de objetos [6] e até de geração de imagens [7], sendo que na literatura existem trabalhos que geram modelo CNN para classificar imagem pornográfica [8-13] e relatam resultados promissores

Para atingir este objetivo, de modo geral, um vídeo é dividido em um conjunto de *frames* (sequência de imagens/fotos) e cada imagem é classificada individualmente [14]. Entretanto, um único vídeo pode ser composto por várias imagens, considerando que elas são tipicamente codificadas na proporção imagens/segundo de 23,97 [15]. Adicionalmente, faz-se lembrar que é normal o uso de CNNs imaginando-se que todo o conjunto de imagens esteja disponível para detecção. No entanto, os vídeos são frequentemente vistos continuamente de uma forma baseada em *streaming* [16].

Para gerar modelo CNN é necessário um treinamento a partir de uma base. Todavia, faltam bases de pornografia publicamente disponíveis, fazendo com que pesquisas na área sejam dificultadas. Aliás, podem existir problemas caso uma base de pornografia esteja pública. Pode-se incorrer em questões legais, já que a divulgação de pornografia é reprimida em alguns países.

Além do que, pode-se corromper a confidencialidade de uma base de pornografia, já que, diante das questões legais, este tipo de informação deve ser restrito apenas a usuários autorizados, em caso contrário a base poderia ser utilizada para disseminação de pornografia. Assim, depara-se com o problema de treinar um detector de pornografia a ser utilizado por usuário e permitir que pesquisadores posteriormente possam realizar *benchmark* com o mesmo.

Também, as CNNs costumam exigir muita memória, *Central Processing Unit* (CPU) e *Graphics Processing Unit* (GPU). Como resultado, modelos CNN não podem ser executadas em tempo real em dispositivos com recursos limitados. Ainda que, normalmente os modelos

[6] https://www.cgi.br/media/docs/publicacoes/2/20200707094721/tic_empresas_2019_livro_eletronico.pdf

são aplicados em ferramentas (sistemas aplicativos), que devem ser *user-friendly*. Por exemplo, durante a visualização de vídeo *online*, um detector de pornografia pode ter seu uso minimizado, propiciando ao usuário uma experiência agradável. Por exemplo, um detector de pornografia pode acusar um falso positivo (FP) e com isso, fazer o bloqueio da visualização do vídeo. Um erro comum onde o usuário deixa de fazer a visualização de um conteúdo que não contêm pornografia. Portanto, é necessário fornecer o menor impacto de visualização na ocorrência de FP (*user-friendly*). Assumindo que conteúdos pornográficos podem ser inseridos em qualquer outro vídeo.

Na sequência apresenta-se a motivação para realização deste trabalho.

1.2. Motivação

A motivação para a realização deste trabalho está em ofuscar as partes íntimas em imagens pornográficas. Para tal, destacam-se quatro problemas: limitações computacionais, falta de bases de pornografia publicamente disponíveis, uso em dispositivos com recursos limitados e ter a característica *user-friendly*.

As CNNs possuem limitações computacionais. Elas exigem muitos recursos computacionais [17]. Por exemplo, uma arquitetura CNN amplamente usada pode exigir até 16 Gigabytes de memória RAM enquanto processa apenas 4,26 imagens/segundo em uma GPU com muito poder de processamento [18]. Consequentemente, a implementação de CNN no dispositivo do usuário final torna-se uma tarefa desafiadora, considerando que as abordagens de detecção disponíveis na literatura geralmente não consideram a tarefa de implantação da CNN, ignorando a falta de recursos disponíveis no dispositivo do usuário final [19].

Ademais, existem dificuldades para realizar pesquisas na área de detecção de pornografia pela falta de bases públicas. Durante formalização do método percebeu-se que a grande maioria dos trabalhos relacionados a pornografia não podem ser comparados, haja vista que a maioria das bases utilizadas não são públicas. Isso dificulta a realização de *benchmark*, assim como incentivo a realização de pesquisas na área.

Por sua vez, vídeos pornográficos são frequentemente visualizados em dispositivos com recursos limitados e existe tendência do aumento dessa prática. Tais dispositivos normalmente não são equipados com *hardware* avançado, como CPU/GPU ou se a possuem não tem os recursos necessários na maioria dos casos. Eles se baseiam no uso de aplicativos

específicos, ou mesmo, aplicativos que não estão presentes no dispositivo. Portanto, quando um usuário de dispositivo com recursos limitados utiliza um aplicativo que faz a detecção de pornografia será demandado que cause o menor impacto possível para o usuário (que seja *user-friendly*).

O usuário, por muitas vezes, deixa de realizar determinadas operações que exigem recursos computacionais, como por exemplo, o uso de serviço de banco de dados alocado no dispositivo. Como resultado, a detecção de pornografia baseada em CNN precisa ser adequada para implantação em tais dispositivos, considerando que, devido às altas demandas computacionais, as CNNs frequentemente requerem *hardware* dedicado, altamente especializado e de alto custo [20].

Ainda no contexto de uso *user-friendly*, verifica-se que um erro comum de classificação são os FP, ou seja, a classificação de conteúdo próprio como impróprio. Neste caso, o usuário deixa de fazer a visualização de conteúdo. Entretanto, no caso de um vídeo, o detector pode finalizar a visualização do vídeo, e se o detector for corretamente configurado, o usuário deixará de visualizar uma porção de imagens (*frames*) consideradas FP.

1.3. Hipótese

Este trabalho possui a seguinte hipótese: detecção *online* e *user-friendly* de pornografia em vídeo sendo visualizado em dispositivos com recursos limitados pode ser efetuada com base em CNN treinada com base ofuscada.

Ao longo dos últimos anos, cresceu o uso de aplicativos que funcionam *online*, como é o caso das redes sociais. Acompanhando este crescimento, usuários têm sido condicionados, no mínimo, a duas situações: acessar conteúdo *online* e através da utilização de ferramentas específicas. Cita-se como exemplo o acesso a *streaming* de vídeo (serviços da Netflix[7], por exemplo), que é feito através de ferramenta.

A primeira situação refere-se ao acesso *online*. É notório e pesquisas confirmam que a cada dia, usuários ficam mais tempo conectados na Internet, ou seja, usuários passaram a

[7] https://www.netflix.com/br/

utilizar dispositivos com recursos computacionais limitados e menos complexos. Por exemplo, muitos usuários não necessitam de dispositivos com recursos para armazenamento avançados, já que a alocação do material está na Internet. Entretanto, a questão de segurança tornou-se preocupante, principalmente quando os usuários forem crianças.

Ademais, em se tratando de ferramentas, elas precisam ter boa usabilidade para que sejam efetivamente utilizadas por usuários. Assim, assume-se necessitar de ferramenta (detector de pornografia em vídeo) que propicie a melhor experiência para o usuário.

A ferramenta *user-friendly* é advinda da percepção de que os atuais detectores de pornografia não se preocupam com o impacto do FP, pois baseiam-se no uso de classificadores de imagem. Na prática, quando um classificador entende um *frame* como pornográfico, não permite a sua visualização.

As ferramentas existentes e que realizam a detecção de pornografia em vídeo ignoram o quesito de *uso amigável*, proporcionando impacto negativo para o usuário.

Verifica-se ainda que pesquisas na área de detecção de pornografia são dificultadas pela falta de bases publicamente disponíveis, visto que elas poderiam ser facilmente compartilhadas na Internet, fomentando a pornografia. Entretanto, existe a possibilidade de ofuscar uma base de pornografia através da inserção de ofuscamento nas partes íntimas em imagens, sendo que este procedimento faz com que usuários percam o interesse pornográfico nesta nova base (base ofuscada). Assim, a base ofuscada poderia ser utilizada por outros pesquisadores, principalmente para realização de *benchmark*.

Os trabalhos acadêmicos da literatura que objetivam a detecção de pornografia, baseados em CNN, e apresentam resultados promissores. Entretanto, eles não são aplicados no mundo real (sociedade), pois se tem a condicionante de que a CNN tenha um custo computacional muito alto.

Há a necessidade do desenvolvimento de uma ferramenta *user-friendly* que, detecte pornografia *online*, funcione em dispositivos com recursos limitados, e que seja baseado em CNN, sendo este último treinado com a base ofuscada, comprovando a possibilidade de detectar pornografia.

1.4. Objetivos

Objetiva-se um método dividido em duas etapas. A primeira concebe um mecanismo que realiza detecção *online* de imagem de pornografia em vídeo no uso de detecção de objetos. A segunda etapa formaliza uma base de imagens de pornografia sem revelar as partes íntimas e comprova a detecção de pornografia mediante o uso de classificação de imagens (CNN).

Os objetivos específicos são:

a. Formalizar e construir uma base de imagens de pornografia não ofuscadas (Pu).
b. Prover um método de detecção de pornografia baseado em detecção de objetos (CNN).
c. Prover um método para ofuscação das partes íntimas, e ao mesmo tempo, reconstituição das partes ofuscadas.
d. Formalizar e construir uma base de imagens de pornografia ofuscadas, i.e., censuradas (Pt).
e. Prover um método de detecção de pornografia baseado em classificação de imagens (CNN) e que permita o *benchmarking* sem expor publicamente as partes íntimas na base.
f. Desenvolver os protótipos.
g. Avaliar a abordagem proposta.

1.5. Contribuições

As principais contribuições do trabalho são listadas a seguir:

- É fornecido uma base publicamente disponível[8] com quatro partes privadas anotadas de um corpo humano (objetos), denominada *Private Parts Object Dataset* (PPO). A base foi criada com inspeção manual de vídeos pornográficos, resultando em 52.215 objetos anotados. O PPO é o primeiro conjunto de dados publicamente disponível na literatura com partes privadas anotadas.

[8] https://secplab.ppgia.pucpr.br/?q=ppcensor

- Apresenta-se uma nova técnica de detecção de conteúdo pornográfico que trata a detecção de pornografia como um problema de detecção de objeto (CNN). A abordagem proposta, a primeira na literatura, é capaz de realizar a detecção de partes íntimas com taxas de precisão semelhantes em comparação com as abordagens do estado da arte tradicionais baseadas em imagens.
- Desenvolvimento de mecanismo (subproduto) para a detecção inteligente e *online* de imagens de pornografia em vídeo disponibilizado em URLs. Ou seja, apresenta-se uma nova arquitetura de detecção baseada em CNN: *Private Parts Censor* (PPCensor) que é adequada para dispositivos com recursos limitados. A arquitetura implementa o processo de detecção de objetos proposto em um servidor *proxy* que roda numa arquitetura paralela para resposta *online* e processamento de *streaming* de vídeo integrando a GPU. Portanto, o usuário não precisa fazer modificações e nem necessita de processamento adicional em seu dispositivo, já que a detecção é executada de forma remota e transparente para o usuário final.
- Método para publicação de imagens de pornografia sem revelar as partes íntimas nas cenas (PPOBench). Faz-se a reconstituição das partes ofuscadas da PPO usando uma *Generative Adversarial Network* (GAN) e uma CCN com o estágio de *Pooling* modificado para permitir o *benchmarking* sem expor publicamente as partes íntimas na base.
- Apresenta-se uma nova técnica de detecção de conteúdo pornográfico baseado em classificação de imagens (CNN). A abordagem proposta é capaz de realizar a detecção de pornografia quando o treinamento é realizado com imagens de pornografia ofuscadas.

1.6. Publicações

- *Periódico Elsevier Future Generation Computer Systems* (2020) [21]: Este artigo propõe o *Private Parts Censor* (PPCensor) como uma arquitetura baseada em CNN para detecção transparente, quase em tempo real e ofuscação de regiões de *frame* de vídeo pornográfico. Qualis A2.
- Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2019) [22]: Neste trabalho, a ausência de integridade na CNN é verificada usando GAN.

Para isso, foi modelado um classificador de autenticidade baseado no algoritmo NB (*Naive Bayes*). Qualis A4.

- Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2020) [23]: Este artigo apresenta a ferramenta PPCensor para detecção de objetos de natureza pornográfica implementado como um *proxy*. Qualis A4.
- Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2020) [24]: Este artigo propõem uma abordagem baseada em aprendizagem por reforço e avaliação da confiabilidade das classificações para a detecçao de intrusão em rede. Qualis A4.
- Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2021): Artigo submetido em 11/06/2021. Título: "Detecção de Mídias Pornográficas em Dispositivos com Recursos Limitados para Controle Parental". Qualis A4.
- *Periódico Elsevier Neurocomputing ou Ieee Transactions On Information Forensics and Security*: Este artigo esta em elaboração e tem data prevista para submissão no mês de outubro de 2021. Título: "*PPOBench - Method for publishing pornography images without revealing the intimate parts of the scenes*". Qualis A2.
- *Periódico Elsevier Future Generation Computer Systems:* O artigo ("*On Real-Time Video Detection in Resource Constrained Devices: a Lightweight CNN-based approach*") esta em elaboração e tem data prevista para submissão no mês de setembro de 2021. Qualis A2.

1.7. Patente

Este trabalho tem:

- Um depósito de patente no Instituto Nacional da Propriedade Industrial[9] (INPI) com o título de “Método de Detecção de Pornografia em Vídeos e de Publicação de Imagens sem Exposição de Partes Íntimas”. A patente de invenção pertence ao campo técnico de métodos de detecção de pornografia envolvendo CiberSegurança com controle e processamento de Imagens/Vídeos em tempo real. Mais especificamente é proposto um método para detecção e ofuscação, em tempo real, de partes íntimas em *streaming* de vídeo *online*, por meio de um PPCensor, aplicando a técnica de detecção de objetos usando *Convolutional Neural Network* (CCN) para classificar e censurar, em tempo real, apenas as partes íntimas nas cenas de um vídeo, assim como para publicação de imagens de pornografia sem revelar as partes íntimas nas cenas, por meio de um PPOBench, realizando a reconstituição das partes ofuscadas da PPO usando uma *Generative Adversarial Network* (GAN) e uma CCN com o estágio de *Pooling* modificado para permitir o *benchmarking* sem expor publicamente as partes íntimas na base. Data do depósito: 16/06/2021.
- Um depósito de *software* com o título: PPCensor: Detecção de Pornografia em Tempo Real no *Streaming* de Vídeo. Data de criação do *software*: 16/07/2020. Data do registro do *software*: 16/06/2021.

1.8. Organização

O documento está organizado da seguinte forma: No Capítulo 2 descreve-se a fundamentação teórica. No Capítulo 3 apresenta-se o problema. No Capítulo 4, os trabalhos relacionados. No Capítulo 5 apresenta-se o método proposto que para melhor explicação foi dividido em duas etapas. No Capítulo 6, os resultados obtidos, e por fim, no Capítulo 7 relata-se as conclusões.

[9] https://www.gov.br/inpi/pt-br

Capítulo 2

Fundamentação

Neste Capítulo aborda-se aspectos teóricos que são utilizados pelo método. São apresentados conceitos da *deep learning* assim como explorado a CNN na classificação de imagens e detecção de objetos: funcionamento e seus parâmetros. Outrossim, conceituam-se as GANS, entre elas, a pix2pix e a pix2pixHD. Também se apresenta *framework* que visa a explicação de modelo treinado (*explainable*), métricas de avaliação, técnica de *transfer learning*, *proxy*, e as discussões relacionadas.

2.1. *Deep Learning*

Deep learning baseia-se na utilização de redes neurais [25], sendo considerada uma subárea pertencente a Aprendizagem de Máquina [20] cada vez mais popular [17]. Cita-se o uso de *deep learning* por empresas, tais como Facebook[10], NVIDIA[11] e Google (Projeto AlphaGo[12]).

Uma rede neural recebe entradas, que são processadas em camadas ocultas usando pesos que são ajustados durante o treinamento. Após este processo, tem-se a geração de um modelo que realizará predições. A rede neural aprende por conta própria mediante a extração de características [17].

Salienta-se que as redes neurais foram originadas pela Inteligência Artificial (IA), simulando o processo das principais áreas sensoriais do cérebro humano. Nosso cérebro pode

[10] https://research.fb.com/category/facebook-ai-research

[11] https://www.nvidia.com/en-us/deep-learning-ai

[12] https://deepmind.com/research/alphago

extrair automaticamente a representação de dados de diferentes cenas. Por exemplo, a entrada é a informação recebida dos olhos, enquanto a saída são os objetos classificados.

Existem redes que aplicam *deep learning* para resolver diversos tipos de problemas. Cita-se as algumas delas: *Deep Belief Networks* (DBNs) [26], *Recurrent Neural Network* (RNNs) [27], *Recursive Neural Network* (RvNN) [28], *Recurrent Convolution Networks* (RCNs) [29], GANs [30] e as CNNs [31-32], sendo que as duas últimas são aplicadas com maior enfoque no atual trabalho.

2.2. CNN

As CNNs possuem várias finalidades, entre elas, a classificação de imagens e a detecção de objetos [6]. Sendo assim, a CNN [31-32] tem sido a rede mais utilizada para o reconhecimento de imagem, classificação e detecção de tarefas, e tem amplo crescimento em vista dos avanços nos recursos computacionais, quantidade volumosa de conjuntos de dados rotulados, e desenvolvimento de algoritmos [5].

Esta rede visa aprender a resolver um problema via treinamento, onde as informações percorrem uma direção e assim geram um modelo que fará predições. Apresenta-se na Figura 1 uma rede padrão da CNN que é composta das camadas: *Convolution* (Conv), *Pooling* e *Full Connected* (FC) *Layers*.

O funcionamento da CNN se baseia no fornecimento de uma grande quantidade de amostras. Elas são fornecidas na *entrada*, e após as imagens são representadas em matrizes de *pixels* que são submetidas a operações matemáticas, que envolvem convoluções e *Poolings*. Na sequência, os resultados são fornecidos a uma ou mais FCs. Assim, realiza-se a produção do rótulo [33].

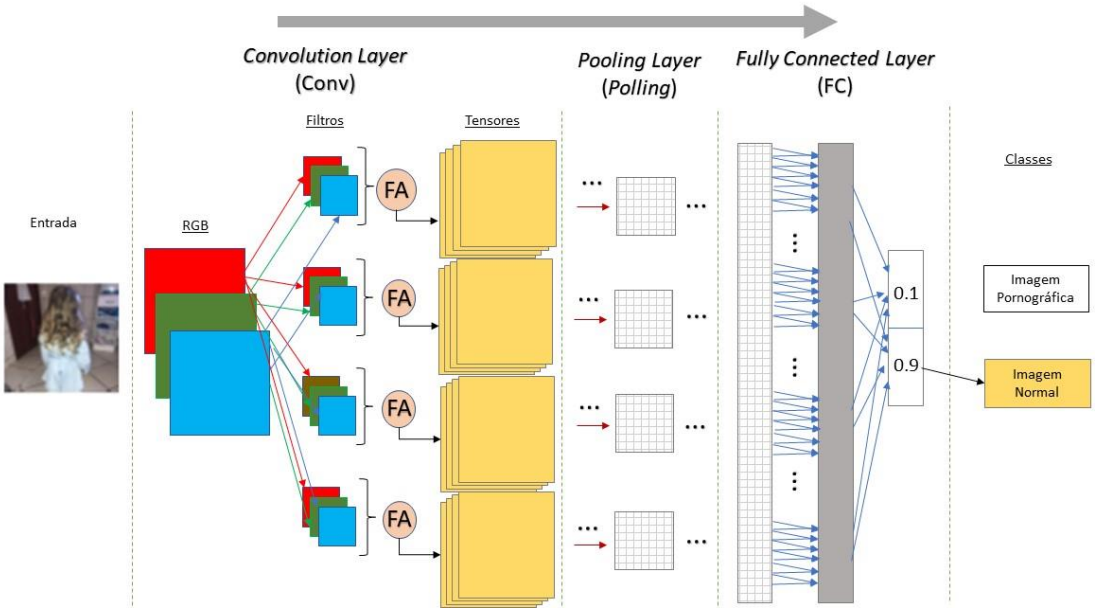


Figura 1. Rede padrão da CNN. Adaptado de [34].

O funcionamento da Conv é baseado em filtro(s), ou *kernels* que são aplicados nas imagens processadas pela CNN [32]. Operações matemáticas denominadas de convolução são aplicadas no conjunto de *pixels* que representam a imagem com os *pixels* que representam o filtro, respeitando-se o *stride*, ou seja, o tamanho do passo com que o filtro é aplicado na imagem. Posteriormente, o resultado da operação produz um *feature map* (mapa de características) em formato de matriz. Por sua vez, o mapa normalmente possuirá dimensões menores que a imagem.

Em função do mapa possuir dimensões menores que a imagem que sofreu o processo de convolução, de forma facultativa pode-se fazer o preenchimento (*padding*) no mapa, para então ser aplicado a função de ativação (FA) no mapa que tem suas dimensões similares a da imagem processada pela Conv [34]. Com isto, permite-se a extração de características, sendo que a FA pode ser utilizada em conjunto com outras camadas. Cita-se como exemplo a função *Rectified Linear Function* (ReLU) [35].

Na sequência, o resultado da ativação (um tensor), é fornecido à camada posterior. No caso da Figura 1, o *Pooling* reduzirá a dimensão da imagem, entre outros, mediante a aplicação de operação matemática [36-37]. As operações mais utilizadas são a *Average* ou *Max Pooling* [5]. Estas operações (cálculos) descartam parte da informação de entrada considerada irrelevante para o resultado. Na Figura 2 exemplifica-se o funcionamento das duas operações

citadas, onde a operação *Average Pooling* resulta na média de um conjunto de *pixels*, e o *Max Pooling* proporciona o maior valor dentre um conjunto de *pixels*.

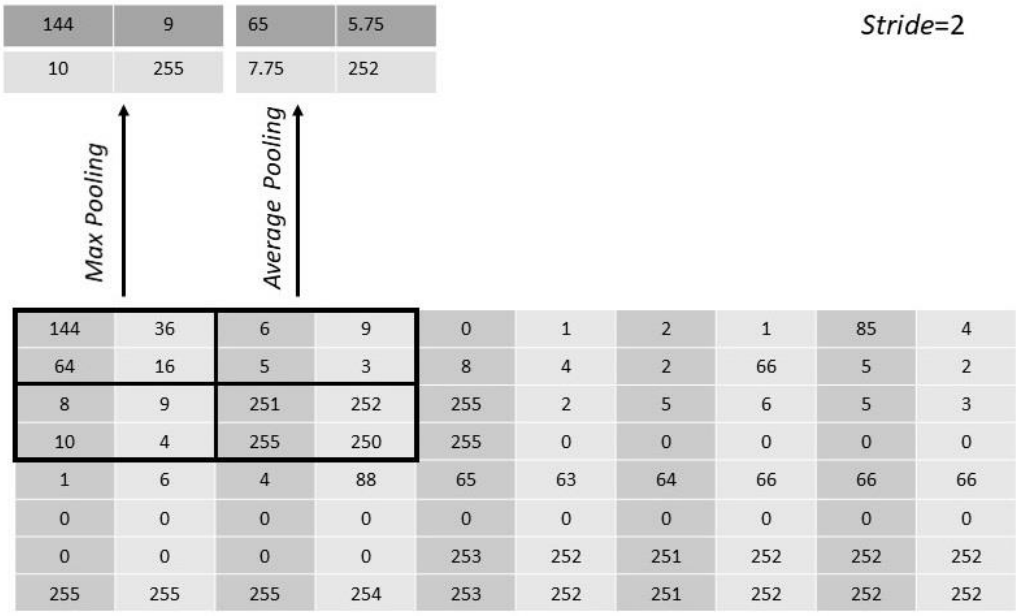


Figura 2. Funcionamento do *Pooling*.

Ainda na Figura 1, todos os pesos aplicados na CNN são enviados para a FC, que representará os pesos linearmente, assim como os valores dos tensores resultantes em um vetor unidimensional. Aplica-se na sequência uma função matemática (por exemplo, o algoritmo *softmax* [31]) entre os pesos, os valores, e o *bias* (conhecimento), produzindo um vetor com as probabilidades [5,34].

Cita-se o caso de uma CNN que gera um mapa de características nas dimensões 4 x 4 x 40. A camada FC transformará os valores desse mapa em um vetor de 1 x 640. Todos os 640 valores serão associados a cada neurônio, produzindo uma combinação linear.

O treinamento de uma CNN envolve duas etapas. A primeira formaliza o modelo e se utiliza dentre outras, de um conjunto de amostras para validação, e algoritmos para ajustar seus parâmetros. Cita-se o algoritmo mais utilizado: *backpropagation* [32,38,40-41]. E durante este treinamento realiza-se a predição. Caso o treinamento seja realizado com 20 épocas, por exemplo, tem-se a geração de 20 modelos onde se usam as imagens da validação. Por sua vez, na segunda etapa deve-se optar pelo melhor modelo gerado, e este modelo será utilizado para uma segunda predição [20], onde se utiliza do conjunto de amostra de teste. É utilizada a

probabilidade de determinada imagem pertencer a classe sendo testada. Assim, quanto maior a probabilidade, maior a chance de uma imagem pertencer aquela classe.

O classificador treinado pode predizer novos objetos automaticamente. E para avaliação do desempenho do classificador, pode-se utilizar diferentes métricas, pois necessita-se verificar a precisão da classificação resultante. Na métrica, a avaliação do desempenho de um modelo de classificação baseia-se na taxa de categorização correta e incorreta do mecanismo. Esses valores são anotados e apresentados em uma matriz de confusão. Assim, é possível realizar a comparação entre diferentes classificadores na solução de um mesmo problema.

Existem algoritmos chamados de *explainable* que visam explicar a predição realizada por modelo treinado [42-43], permitindo que seres humanos tenham melhor compreensão dos resultados. O *Layer-wise Relevance Propagation*[13] (LRP) faz explicação atribuindo pontos de relevância a componentes importantes da entrada usando o modelo do usuário [42], e este algoritmo é compatível com o *framework* Caffe[14].

Ademais, na literatura, uma abordagem comum para melhorar a precisão da CNN é aumentar a complexidade da rede. Por exemplo, vários trabalhos consideraram aumentar o número de camadas, aumentando assim as demandas de processamento e o tamanho da memória. Para implantar uma técnica de classificação baseada em CNN, é comum contar com *hardware* dedicado, equipado com GPUs de ponta. No entanto, a implantação de tais abordagens é um desafio para dispositivos com recursos limitados.

2.2.1. Detecção de Objetos

As técnicas tradicionais baseadas em CNN classificam uma determinada imagem de entrada como pertencente ou não a uma classe (por exemplo, pornografia ou normal). No entanto, em algumas aplicações, uma determinada imagem pode conter várias regiões de

[13] https://github.com/sebastian-lapuschkin/lrp_toolbox

[14] https://caffe.berkeleyvision.org/

interesse. Um exemplo disso é um sistema projetado para detecção de pessoas em que uma única imagem de entrada pode conter várias pessoas para identificação.

Nesse caso, várias regiões da imagem devem ser identificadas em oposição à classificação da imagem inteira. Consequentemente, abordagens baseadas em CNN também têm sido usadas para tarefas de detecção de objetos em imagens [44].

O objetivo é identificar uma região da caixa delimitadora (região da imagem) que inclua corretamente a ocorrência do objeto na imagem. Várias abordagens têm sido propostas para atingir este objetivo, que difere dos extratores de características para o tamanho do teste de imagem de entrada [45]. Citam-se os detectores *Faster R-CNN* [46], *Single Shot Detector* (SSD) [47] e *You Only Look Once* (YOLO) [48]. Semelhante às técnicas de classificação, os detectores de objetos também são técnicas baseadas em treinamento.

Existem duas características principais no *Faster R-CNN* [46]. Primeiramente, os recursos são extraídos das imagens e enviados para a *Region Proposal Network* (RPN), que realiza a extração dos recursos de acordo com as imagens de teste. Em segundo lugar, a detecção é realizada, ou seja, para cada local, a probabilidade de existência do objeto pesquisado é calculada.

A arquitetura SSD é executada em apenas uma etapa, visto que as SSDs são consideradas rápidas e adequadas para *hardware* com recursos limitados [45], como telefones celulares e tablets. Para melhorar a precisão, os pesquisadores costumam utilizar uma arquitetura CNN mais complexa, como ao *Faster R-CNN*.

Semelhante à classificação baseada em imagens, é possível usar *transfer learning* [49-50] para treinar detectores de objetos. Como tal, esses modelos treinados facilitam o refinamento de pesos usando CNNs pré-treinadas. Em geral, *transfer learning* na detecção de objetos é realizada por meio da base de imagens COCO [51], que consiste em 330.000 imagens divididas em 80 classes (objetos). Em geral, os detectores de objetos baseados em CNN são avaliados em relação à sua taxa de transferência de detecção e precisão média (mAP). O mAP é calculado de acordo com a interseção sobre a união (IoU). Em geral, na literatura, é aplicado um limite de 0,5 para cada caixa delimitadora de objeto. O IoU estabelece a região da caixa delimitadora identificada que cruza com a região do objeto de destino [44].

2.2.2. Parâmetros

As CNNs são consideradas complexas. Existe a dificuldade de compreensão do relacionamento de seus parâmetros, embora o número total se reflita no número de filtros aplicados, *bias* e neurônios disponibilizados [5,14,34]. O número de parâmetros é relacionado ao número de pesos que a CNN precisa aprender. Exemplifica-se pela Figura 3 uma arquitetura CNN composta pelas camadas: três Convs (Conv 1, Conv 2 e Conv 3), dois *Poolings*, e de duas FCs (FC1 e FC2).

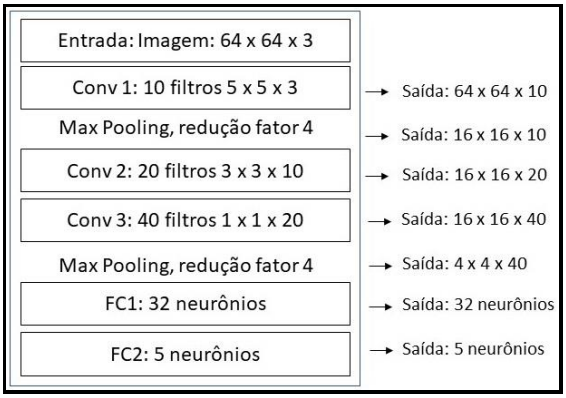


Figura 3. Exemplo de rede CNN. Adaptado de [36].

O cálculo da quantidade de parâmetros para a rede apresentada na Figura 3 é realizado em duas etapas. Na primeira, calcula-se para cada Conv (Equação 2.1) a multiplicação das dimensões do filtro aplicado por determinada Conv (altura x comprimento x profundidade) e soma-se o resultado desta multiplicação ao *bias*. Após, aplica-se a multiplicação do valor resultante à quantidade de filtros da Conv sendo analisada.

$$Total_{Conv} = (Num_{Filtros}\ x\ [dimensões\_filtro + bias]) \qquad (2.1)$$

A segunda etapa envolve o cálculo de parâmetros para as FCs. Multiplica-se a quantidade de neurônios de cada FC pelas dimensões do vetor formalizado. E por fim realiza-se a soma dos parâmetros individualmente calculados, conforme Tabela 1.

Tabela 1. Resultado – quantidade de parâmetros. Adaptado de [34].

$(10\ x\ [5\ x\ 5\ x\ 3 + 1] = 760\ [Conv1]$
(20 x [3 x 3 x 10 + 1] = 1820 [*Conv2*]
(40 x [1 x 1 x 20 + 1] = 840 [*Conv3*]
(32 x [640 + 1] = 20512 [*FC1*]
(5 x [32 + 1] = 165) [*FC2*]
TOTAL = 760 + 1820 + 840 + 20512 + 165 = 24097

2.3. GANS

As GANs compreendem um par de redes neurais [30]. Elas aprendem a copiar uma distribuição de dados das informações de entrada e com isso gerar nova informação. Para isso, se utilizam de um gerador (*generator*) e um descriminador (*discriminator*). O potencial do uso de GAN é enorme, visto que ela pode imitar qualquer distribuição de dados, i.e., gerar imagem falsa [52]. Elas têm sido usadas com sucesso em várias aplicações, como geração de imagens sintéticas e pintura de imagens [53], gerando imagens mais nítidas e aprimoradas. Ademais, existem vários modelos, entre elas, pix2pix [7] e a pix2pixHD [54].

Pix2pix[15] é uma implementação de translação de imagem baseada em redes condicionais adversárias [7], onde se aprende como converter uma imagem em outra. O aprendizado ocorre através de uma função de perda. Após a pix2pix, foi originada a pix2pixHD [54].

A segunda é aprimoramento da primeira: diferenças na estrutura do gerador, arquitetura do descriminador, e na função de aprendizagem [54]. Por exemplo, na pix2pix, as imagens são reduzidas para dimensões 256 x 256, enquanto na outra, 2048 x 1024, sendo possível configurações menores. Ressalta-se que quanto maior as dimensões, melhores devem ser os recursos da GPU utilizada no treinamento.

[15] https://github.com/phillipi/pix2pix

2.4. Discussão

Neste Capítulo foram abordados conceitos de *deep learning*, CNN (classificação de imagens e detecção de objetos), exemplo de utilizações, alguns de seus parâmetros e o cálculo do número de parâmetros de uma CNN. Ademais, foram abordados as GANs e algoritmo para explicar modelos treinados (*explainable*).

Dentre as CNNs de maior destaque ressalta-se as arquiteturas denominadas: ResNet [55], InceptionV3 [56] e VGG-Net [57], sendo as arquiteturas VGG-16 e VGG-19 as mais comuns. Na implementação da arquitetura CNN, é necessário a inicialização de parâmetros de configuração, como por exemplo, número de épocas (*epochs*), taxa de aprendizado (*learning rate*), *decay* e *momentum* [34]. O número de camadas de uma CNN pode variar, sendo o mais usual a utilização de uma Conv, e após uma *Pooling*.

Em específico, na camada de *Pooling* são executadas operações matemáticas, como *Max* ou *Average Pooling*. Nesta camada também poderiam ser aplicadas outras operações, como por exemplo, o *gamma correction*. Em processamento de imagens, esta função pode aplicar várias operações, dentre elas, a função de raiz quadrada para iluminar uma imagem [58]. Desta forma, é possível o uso parcial de *gamma correction* em camada da CNN, como no *Pooling*. Permite-se que as imagens processadas pela CNN possam ser manipuladas ao longo do processamento da arquitetura, ou seja, melhorando as características que são extraídas pela CNN.

É possível a utilização de arquiteturas CNN com pesos inicializados aleatoriamente, ou pré-treinados (*transfer learning*). No caso de *transfer learning* [49-50], pode-se utilizar os pesos de um treinamento já realizado ou a extração de características fixado na rede. Cita-se o uso de pesos do treinamento ImageNet que possui mais de 1.2 milhões de imagens divididas em 1000 classes [31].

Quando se treina uma CNN objetiva-se um modelo para classificar um problema, podendo este se relacionar com diversas áreas, i.e., interdisciplinar. Segurança Computacional é uma delas, e tem o objetivo da criação de normas que visam a proteção de usuários contra ameaças [59-60]. No caso de CNNs, a Segurança Computacional pode se utilizar de GANs para evitar que se confundam imagens normais como sendo imagem pornográfica por exemplo. As GANs são modelos generativos, e tem o intuito de produzir uma saída complexa [52].

Existe um dispositivo de segurança que pode filtrar, entre outros, os pacotes que trafegam em uma rede de computadores: o *proxy*. Este mecanismo possui vários sinônimos, e um deles é *cache* Web [61]: responsável por intermediar as requisições Web de usuários. A exemplo cita-se a requisição de objetos disponibilizados em determinada URL.

Para usuários utilizarem os serviços de um *proxy*, precisam a ele estar conectado. De forma semelhante, o *proxy* precisa estar conectando a Web. As requisições de usuários são direcionadas ao servidor *proxy*, i.e., este dispositivo pode ser caracterizado como servidor e cliente simultaneamente, pois tem acesso a todo o conteúdo da Web solicitado por um cliente [62].

Por sua vez, uma imagem pode estar contida em um objeto solicitado a um *proxy*. Ela é tratada via Processamento de Imagens Digitais e é definida como aquisição e processamento informático de informações visuais [63-64]. As imagens possuem características como: dimensões, espaço de cores, tonalidades e resolução. Pode ter sua origem a partir de *frames* de um vídeo por exemplo [64], sendo que quanto melhor a qualidade do vídeo, maior a quantidade de *frames* por segundo.

Diante do exposto, foi apresentado teoricamente neste Capítulo os principais tópicos a serem utilizados neste trabalho. No próximo Capítulo detalha-se os trabalhos relacionados ao objeto de estudo que se apresenta.

Capítulo 3

O Desafio da Detecção *Online* de Vídeos Pornográficos

Neste Capítulo é feita a avaliação do desempenho das abordagens de detecção disponíveis publicamente para classificação de conteúdo pornográfico e analisado a acurácia (ACC) proporcionada por CNN que detecta imagem de pornografia baseando-se em treinamento com imagens pornográficas ofuscadas (censuradas). Especificamente, primeiro descreve-se as bases utilizadas. Em seguida, avalia-se as ferramentas NuDetective [65] e o Yahoo Detector [9] quanto à ACC e a taxa de transferência (*throughput*). Na sequência, avalia-se a ACC proporcionada por duas técnicas de ofuscamento: detector de objetos baseado em CNN [44] contrário ao ofuscamento manual, tendo como *baseline* as imagens não ofuscadas. As avaliações foram executadas em um computador *desktop* equipado com CPU Intel I7 (quad-core), 16 GB de memória e GPU Nvidia Titan-XP.

3.1. Bases Utilizadas

O método foi desenvolvido no uso de *frames* (imagens**)** extraídas da base de vídeos Pornography-2k [66] e UCF101 [67]. Pornography-2k é composta por 2.000 vídeos, dos quais metade são pornográficos e a outra, normais, i.e., se dividem em duas classes: pornográfico e normal. Exemplos de imagens da Pornography-2k são apresentadas na Figura 4. Por sua vez, UCF101 [67] consiste em 13.320 vídeos normais (possui uma classe) e exemplos são visualizados na Figura 5.

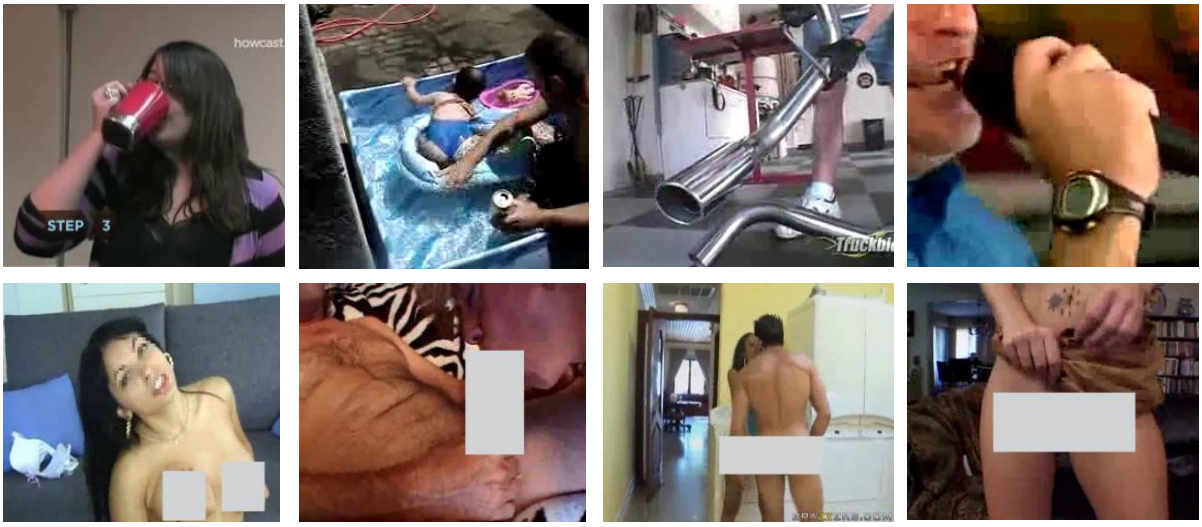


Figura 4. Exemplos de imagens da base Pornography-2k [66]. Na prática, as imagens pornográficas não estão ofuscadas. Elas foram censuradas para efeitos de edição.

Todos os vídeos tiveram suas imagens extraídas em um intervalo de 10 imagens (aproximadamente a cada 0,3 segundos) no uso da API OpenCV[16]. Portanto, as imagens foram extraídas para análise posterior. Os vídeos, e consequentemente suas imagens, foram divididos na proporção de 60/20/20 para efeitos de treinamento, validação e teste. As imagens do treinamento são usadas para fins de aprendizado; de validação para o processo de ajuste do treinamento do modelo; e o teste, para a avaliação final do modelo.

Figura 5. Exemplos de imagens da base UCF101 [67].

No total, a base Pornography-2k formalizou 1.376.035 imagens, nos quais 389.808 são normais e 986.233 possuem conteúdo pornográfico. A base inclui a presença de indivíduos de várias etnias envolvidos em diferentes atividades e com várias qualidades de vídeo, sendo que

[16] https://opencv.org/

os vídeos pornográficos possuem apenas presença de adultos. A base UCF101 [67] formalizou 240.810 imagens normais.

3.2. A Confiabilidade dos Esquemas de Detecção de Pornografia

Duas abordagens de detecção de pornografia, Yahoo Detector [9] e NuDetective [65], foram avaliadas nas bases apresentadas. A Tabela 2 mostra os resultados para Yahoo Detector e NuDetective em relação à taxa de transferência e ACC de detecção. Pode-se observar que o NuDetective supera significativamente o Yahoo Detector em termos de taxa de transferência, atingindo até 411 *frames*/segundo, em contraste com 3,67 *frames*/segundo.

No entanto, o Yahoo Detector melhora significativamente a ACC de detecção, apresentando 45,68% menos FP para Pornography-2k e 21,64% menos para UCF101 quando comparado com NuDetective. Em contraste, com relação às taxas de falsos negativos (FN), o NuDetective supera o Yahoo Detector em 2,47%. Isso ocorre porque o NuDetective realiza a detecção com base no tom da pele. Como resultado, a taxa de transferência de detecção é significativamente melhorada e as taxas de FP também são mais altas em comparação com uma abordagem baseada em CNN.

Tabela 2. Acurácia e taxa de transferência das abordagens de detecção de pornografia [21].

Técnica de Detecção	Detecção - Taxa de Transferência (*frames* /segundo)	Base		
		Pornography-2k		UCF101
		FP (%)	FN (%)	FP (%)
Yahoo Detector [9]	3.67	12.80	5.53	1.61
NuDetective [64]	411.64	58.48	3.06	23.25

3.3. Detecção de Objetos para o Ofuscamento

O objetivo desta seção é comprovar que o uso da técnica de detecção de objetos baseada em CNN [44] é eficaz para ofuscamento das partes íntimas de imagens de pornografia. Tem-se o objetivo de publicar uma base de pornografia ofuscada, e não se deseja que esta base seja utilizada para fins de divulgação de pornografia.

Para tal, compara-se a técnica de detecção de objetos com o ofuscamento manual em um subconjunto de imagens da Pornography-2k [66]. Elas foram avaliadas no uso do *framework* Caffe[17], CNN CaffeNet [68], *transfer learning* [49-50] baseada na ImageNet [69], *learning rate* 1e-05, 80.000 *steps*, *lr_policy step*, *gamma* 0.1, *momentum* 0,9 e *weight_decay* 0.0005. Ou seja, foi gerado modelo para classificação de imagens (CNN) para cada técnica.

Cada técnica foi experimentada com três bases. Para tal, inicialmente foi formalizada a base *baseline*, composta por 50.000 imagens selecionadas das imagens extraídas da base Pornography-2k [66]. Ela foi originada a partir da seleção manual de imagens divididas entre as classes: pornografia e normal. As imagens foram separadas nas proporções de 60/20/20 (treinamento/validação/teste). Na Figura 6 apresentam-se quatro exemplos de imagens pornográficas da *baseline*. E quatro imagens na Figura 7. Salienta-se que as imagens da Figura 6 foram censuradas para visualização.

Figura 6. Exemplos de imagens pornográficas da base *baseline*. Na prática, as imagens pornográficas não estão ofuscadas. Elas foram censuradas para efeitos de edição.

[17] https://caffe.berkeleyvision.org/

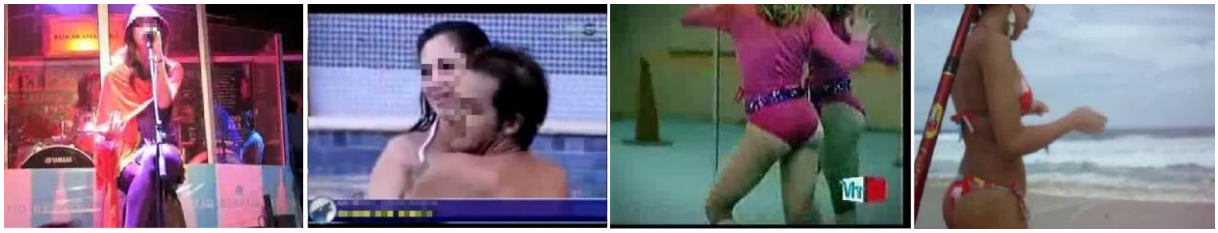


Figura 7. Exemplos de imagens normais da base *baseline*.

Na sequência, deu-se origem as bases Detector e Manual. Elas são similares a base *baseline*. As bases Detector e Manual tiveram ofuscamento somente nas partes íntimas das imagens pornográficas do treinamento/validação. Ou seja, as bases Detector e Manual também possuem duas classes: pornografia e normal.

Para ofuscar a base Detector foi utilizado o detector de objetos gerado no trabalho [21]. Quatro exemplos de imagens pornográficas da base Detector são apresentados na Figura 8. As imagens pertencem a porção de treinamento da classe pornográfica, sendo que as imagens da classe não pornográfica são iguais a da base *baseline* (Figura 7).

Figura 8. Exemplos de imagens da base Detector.

Por sua vez, a base Manual teve ofuscamento realizado manualmente. Neste momento, foi utilizado recursos da API OpenCV, como por exemplo, com a inserção de retângulos no local do ofuscamento. Quatro exemplos de imagens pornográficas retiradas da porção de treinamento são apresentados na Figura 9. As imagens da classe não pornográfica da base Manual são iguais a da base *baseline* (Figura 7).

Figura 9. Exemplos de imagens da base Manual.

Na experimentação foram gerados três modelos. Primeiro utilizando a base *Baseline*. Após Detector, e então, Manual. A Tabela 3 mostra os resultados. O *Baseline* proporcionou 93,73%, Detector 49,91%, e a base Manual, 60,35% de ACC. Com isso, Detector possui 35,61% a menos de ACC em relação ao *Baseline*, e 17,30% a menos que a Manual.

Ademais, percebe-se que a taxa de verdadeiro positivo (TPR) e a taxa de verdadeiro negativo (TNR) para Detector possuem valores menores do que para base Manual. Isso indica que o número de acertos para a base Detector é inferior, ocasionado pela inserção do ofuscamento via detector de objetos. Pode-se considerar que o ofuscamento inserido na base Manual, permite a existência de "evidências" pornográficas. Por sua vez, Manual apresentou 39,66% a menos de taxa de falso positivo (FPR) em relação ao *Baseline*, e 91,26% a menos do que Detector.

Tabela 3. Resultados.

Experimento	Ofuscamento	TPR	TNR	FPR	FNR	ACC (%)
Baseline	Não	0,9587	0,9178	0,0822	0,0413	93,73
Detector	Sim	0,4995	0,4328	0,5672	0,5005	49,91
Manual	Sim	0,5585	0,9504	0,0496	0,4415	60,35

Por fim, foram realizados outros experimentos, onde substituíram-se imagens pornográficas das porções do treinamento e validação das bases Detector e Manual, por imagens pornográficas não ofuscadas. Em diferentes proporções, sendo elas: 5, 10 e 20%. Os resultados da ACC são apresentados na Tabela 4. Quanto maior a quantidade de imagens não ofuscadas inseridas nas bases, maior a ACC apresentada. Entretanto, percebe-se que esta situação muda a partir da inserção de 20%.

Tabela 4. Acurácia da CNN – inserção de imagens pornográficas não ofuscadas.

Quantidade de Imagens Não Ofuscadas (%)	Base	ACC (%)
5	Detector	54,29
5	Manual	60,25
10	Detector	54,66
10	Manual	74,99
20	Detector	77,28
20	Manual	75,89

3.4. Discussão

Na seção 3.2, foram utilizadas as técnicas para a detecção de conteúdo pornográfico em vídeos. Assim, demonstrou-se que as mesmas são incapazes de lidar com a detecção em tempo real, mesmo quando executada em *hardware* dedicado. Essas técnicas apresentam um baixo rendimento ou uma taxa de erro significativamente alta.

Consequentemente, elas não podem ser executadas em dispositivos com recursos limitados, que são ambientes onde o conteúdo pornográfico é frequentemente compartilhado e visualizado, devido à sua baixa precisão ou alta demanda computacional. No entanto, as taxas de ACC obtidas tornam essas técnicas de detecção pouco confiáveis.

Portanto, devido a uma alta taxa de FP, o usuário não confia mais na abordagem de detecção, lembrando que a melhor taxa de FP é obtida usando o Yahoo Detector com 1,61% para a base UCF101. Além disso, como essas técnicas classificam o vídeo avaliado como pornográfico ou não, no caso do FP, o usuário não consegue visualizar o vídeo na íntegra.

É imperativo que as técnicas de detecção de pornografia possam ser implementadas em dispositivos com recursos limitados, realizar detecção em tempo real e abordar adequadamente o desafio do FP de uma maneira *user-friendly*. Isso facilitará o uso confiável e apropriado em ambientes de produção.

Na segunda parte do Capítulo 3 (seção 3.3), foi verificado que o experimento que teve as imagens ofuscadas pelo detector de objetos baseado em CNN obteve uma ACC menor em todas as situações (Tabela 3). Ou seja, o detector de objetos, se bem treinado, consegue detectar e mapear todo o objeto, coisa que o ser humano manualmente não consegue com exatidão. Ainda, percebe-se que o detector de objetos cria um padrão no ofuscamento.

Assim, as partes íntimas são ofuscadas das imagens em exibição sem a necessidade de iteração do usuário. Outrora, comprova-se que a CNN também faz uso da porção da imagem que foi ofuscada. Percebe-se que o detector de objetos proporciona um padrão no ofuscamento, que é resultado de seu treinamento. Ou seja, a primeira conclusão é que a inserção do ofuscamento faz com que a CNN classifique o que tem ofuscamento, e o que não tem ofuscamento.

Por fim, vide Tabela 4, verifica-se quanto maior o número de imagens não ofuscadas presente nas bases, existe mudança na ACC da CNN. Quando foram inseridas 5% e 10% de

imagens não ofuscadas, a base Manual propiciou ACC maior. E quando o número de imagens foi de 20%, a maior ACC foi do detector. Um especialista pode “esquecer” de ofuscar uma parte íntima, e quando isto ocorre, a probabilidade de uma CNN classificar corretamente é maior, tendo como limiar o valor de 15%.

Capítulo 4

Trabalhos Relacionados

Neste Capítulo são apresentados trabalhos divididos em quatro grupos: aplicações baseadas em GAN, detecção de pornografia, utilização de detector de objeto e trabalhos que abordam a proposição de *Pooling.*

4.1. Aplicações GAN

Os autores T. Kim e J. Yang [70] propuseram a rede adversária semigerativa (PPSGAN). Ela é uma metodologia para anonimizar algumas características mantendo a utilidade da base [70]. Substitui-se a privacidade adicionando ruído em imagens e classificadores avaliam a imagem processada por meio da comparação com amostras reais, sendo que é preservado as características originais nas imagens geradas. A avaliação foi realizada no uso das bases: MNIST, Fashion-MNIST, CIFAR-10 e SVHN. O problema está em publicar uma base anonima e que mantenha sua utilidade, ou seja, realização de *benchmark*. A PPSGAN é uma melhoria da rede PPAPNet [71], que se utiliza da ACGAN [72].

M. D. More et al. [73] manifestam que as atuais ferramentas que fazem a classificação de pornografia, normalmente, são configuradas para bloquear imagens pornograficas. Para remediar isso, desenvolveram um método para translação de imagens (*image-to-image translation*) baseando-se na CycleGAN [74]. Faz-se a localização das partes íntimas em imagens, as quais são cobertas por outra imagen (biquíni). Para experimentos, foram utilizadas 1024 imagens de mulheres nuas e 1161 imagens de mulheres vestindo biquíni. Assim, verifica-se que os autores recorrem a método para censurar imagem. Desta forma, usuários não deixaram de visualizar uma imagem, e sim, visualizarão uma imagem censurada. Entretanto, a ferramenta desenvolvida não é aplicável.

Por sua vez, os autores G. S. Simões et al. [75] complementam a problemática anterior que trata da necessidade de ferramentas para censurar imagens pornográficas. Estenderam o

método desenvolvido anteriormente [73], objetivando melhorar a qualidade geral das imagens geradas. Para tal, desenvolveram a AttGAN+ objetivando cobrir o corpo na imagem e classificar (CNN) cinco classes que representam as partes íntimas. Posteriormente, foram realizadas avaliações da qualidade das imagens geradas mediante disponibilização de um questionário aplicado via Internet para 1000 usuários. Reforçando, I. Joshi et al. [76] propôs um algoritmo baseado em CGAN (*Conditional* GAN) para o aprimoramento de impressões digitais latentes. Aplicações policiais e forenses que se baseiam em impressões digitais latentes são muito lentas, e este aprimoramento deixaria os processos mais rápidos. A contribuição deste trabalho encontra-se no desenvolvimento do *Enhanced Fingerprints Reconstruction Loss*. É uma função de geração contínua de imagens, até o mais próximo do *Groundtrugh Binarized Image* (imagem original).

H. Hukkelås et al. [77] apresentam uma arquitetura baseada em CGAN que torna anônimo o rosto detectado em imagens. Assim, denota-se um novo discriminador e gerador. Ademais, para fazer funcionar o modelo é necessária prévia rotulação. Uma anotação de caixa delimitadora para identificar a área sensível (rostos) a ser anonimizada e uma estimativa de pose, contendo itens como: orelhas, olhos, nariz e ombros. Para experimentos, os autores formalizaram a base *Flickr Diverse Faces* (FDF) que consiste em 1,47 milhões de rostos com uma caixa delimitadora e anotação de ponto-chave para cada rosto. Usando FDF treinam o modelo e realizam experimentos com a base WIDER-FACE. Salienta-se que esta base possui 32.203 imagens com 393.703 rostos rotulados. Aplicaram a *Dual Shot Face Detector* (DSFD) [77] na base WIDER-FACE, e os resultados foram registrados com a métrica *Frèchet Inception Distance* [79]. De forma semelhante ao problema apresentado por T. Kim e J. Yang [70], realiza-se a ofuscação, principalmente por questões de direitos autorais (não identificação de rostos).

Atualmente discute-se na sociedade, o problema do vazamento de dados sensíveis. Para tal, aborda-se o uso de GAN para a geração de imagens falsas, com o objetivo de tornar anônimas as informações de usuários do setor saúde [80]. Assim, as imagens falsas podem ser utilizadas em várias áreas, inclusive para pesquisa. Nos experimentos, aplicaram o modelo *Fully Connected Network* (FCN) e CNN nas bases Fingerprints (75.600 amostras) e Iris (2.224 amostras). O resultado da pesquisa é um protótipo de sistema GAN que geram imagens pessoais que imitam as amostras fornecidas. O desempenho do protótipo e resultados foram obtidos baseando-se na visualização das imagens geradas. Dando continuidade a este trabalho, os mesmos autores [81] repetem os experimentos, entretanto com bases diferentes (tireoides e

cardiogramas). Os autores realizam tratamento nas imagens geradas pela GAN. Sendo assim, comparam os resultados proporcionados pelas imagens reais com as imagens produzidas.

4.2. Detecção de Pornografia

Nos últimos anos, diversas abordagens têm sido propostas na literatura visando a detecção de conteúdo pornográfico em vídeos. A divulgação de pornografia se tornou um problema. Por exemplo, foi proposto pelo Yahoo, a saber, Yahoo Detector [9], que é uma ferramenta publicamente disponível para localizar imagens *Not-Safe-For-Work* (NSFW). A técnica aplica uma arquitetura CNN denominada de ResNet50 [55] treinada com *transfer learning* [49-50] baseada na ImageNet [31]. No entanto, embora essas técnicas produzam resultados razoavelmente precisos, a base de imagens utilizada para treinamento não foi compartilhada. Como tal, seus resultados não podem ser comparados.

Em contraste, F. Nian et al. [82] classificou imagem pornográfica usando arquitetura CNN treinada usando *transfer learning* [49-50], e uma base de treinamento composta por 48.600 imagens (13.300 pornográficas e 35.300 normais) obtidas na Internet. Os autores confiaram nas representações de nível médio e no ajuste dos dados de treinamento como duas estratégias principais. A principal contribuição dos autores baseia-se na aplicação do "*fixed-point algorithm*" [83], que permite a detecção de diferentes tamanhos de imagens.

A detecção de conteúdo pornográfico em vídeo ainda está em sua infância. Em geral, as abordagens de detecção são avaliadas usando várias bases, como a Pornography-2k [66], Sensitive [84] e NPDI [85]. Por exemplo, Moustafa [8] aplicou várias arquiteturas CNN incluindo AlexNet [69] e GoogleNet [86] para a detecção de vídeos pornográficos. Em seu trabalho, um vídeo foi categorizado como pornográfico se a maioria de seus *frames* fosse classificada como contendo conteúdo impróprio, conforme avaliado na base NDPI [85].

Em Vitorino et al. [12], a arquitetura CNN (GoogleNet [86]) foi treinada usando *transfer learning* [49-50] e experimentada na base Pornography-2k [66]. Aplicando esta base para uma avaliação e um *benchmark* para comparação de precisão foi fornecido. Por exemplo, o Yahoo Detector atingiu 88% de ACC quando a base foi usada em [12], entretanto a Pornography-2k é uma base de vídeos e não se sabe qual foi o conjunto de imagens utilizado.

M. Perez et al. [10] também avaliaram sua técnica usando a base Pornography-2k [66] e a NPDI [85]. Os autores propuseram o uso de recursos estáticos e de movimento para

classificação de vídeos baseada em CNN. Sua abordagem melhorou significativamente a precisão da detecção, mas não pode ser implementada *online* porque requer o vídeo inteiro, que é necessário para a extração de recursos baseados em movimento. Nessa investigação, o melhor método rendeu uma precisão de classificação de 97,9% no uso da base NPDI [85].

Foi proposto uma arquitetura de aprendizado profundo chamada ACORDE [11]. Sua arquitetura de detecção consiste em CNN e redes recorrentes para a detecção de conteúdo adulto em vídeos. Os experimentos foram realizados usando a base NPDI [85], que produziu resultados razoavelmente precisos, reduzindo assim o número de FPs e FNs em relação às abordagens que utilizaram esse mesmo conjunto de dados.

Zhikang et al. [87] propõem a arquitetura PornNet onde integram-se duas redes para classificação de vídeo pornográfico. Utilizam como treinamento *frames* e áudios extraídos de vídeos pornográficos. No mesmo sentido, Gangwar et al. [88] propuseram conjuntos de dados, os quais são usados para treinamento de CNN para detecção de conteúdo pornográfico e classificação de imagem de pessoa por faixa etária (menor ou adulto).

De forma similar, existem trabalhos não acâdemicos que executam este tipo de detecção. Cita-se a API (*Application Programming Interface*): *The Nudity Detection*[18], ou mesmo outros produtos comerciais, como *Google Vision*, *Amazon Rekognition*[19] e *Microsoft Cognitive Services*.

4.3. Detector de Objetos

Tem sido proposto vários trabalhos que se utilizam de detector de objetos para resolver vários tipos de problemas. Por exemplo, detecção de veículos [89-90], pessoas [91], e até mesmo para análise de evidências de cena de crime [92].

Continuando com o problema de detecção de pornografia, X. Ou et al. [84] apresentam a classificação de imagens/vídeos adultos utilizando a rede multi-contexto profunda (DMCNet)

[18] https://sightengine.com/image-moderation

[19] https://aws.amazon.com/pt/rekognition/?blog-cards.sort-by=item.additionalFields.createdDate&blog-cards.sort-order=desc

[93], um método hierárquico que usa a saída de recursos de um modelo CNN para treinar o RPN da *Faster R-CNN*. Para fins de comparação, os autores geraram a base Sensitive [84] e fizeram comparações com outras bases. No entanto, embora os autores tenham aplicado uma arquitetura CNN de detecção de objeto, eles não detectaram as partes íntimas.

4.4. *Pooling*

Existem muitos trabalhos que propõem operações a serem executados na camada de *Pooling* da CNN, como por exemplo: [94-99]. De tal forma, autores N. Akhtar e U. Ragavendran [37] realizaram um estudo que classificou as abordagens de *Poolings* em quatro categorias: valor, probabilidade, classificação e domínio transformado.

A primeira categoria usa critérios como o tamanho do *kernel*, mapa de recursos de entrada e saída, entre outros. O segundo, tem como base o valor de probabilidade. O terceiro é baseado na atribuição para classificação do peso da ativação. Por fim, os de domínio transformam a imagem em outros domínios. Portanto, serão abordados alguns trabalhos relacionados que se assemelham a proposta (método PPOBench).

Propõem-se o RoI *Pooling* no detector de objetos *Fast Region-based Convolutional Network* (Fast R-CNN) [100], alegando que a velocidade da rede anterior poderia ser melhorada [101]. No funcionamento do detector, e ao se identificar a região das propostas, ele utiliza o RoI *Pooling*. Uma imagem de entrada e várias regiões de interesse são inseridas em uma rede totalmente convolucional. Cada RoI é agrupado em um mapa de recursos de tamanho fixo e, em seguida, mapeado para um vetor de recursos por camadas totalmente conectadas (FCs). Os resultados proporcionados por este detector foram promissores. A precisão de detecção foi melhorada (mAP) e a velocidade de treinamento e teste também. Isso quando comparado a R-CNN.

M. Sun et al. [102] detectam que o desempenho de CNNs pode ser melhorado. Portanto, sugerem uma combinação linear aplicada no *Pooling* chamado de LEAP. Ele aprende um filtro linear através do compartilhamento dos mapas de características (*activation maps*). O filtro linear agrega os recursos em sua região operacional. A convolução simplificada reduz muito o número de parâmetros, mantendo um desempenho comparável. Foram realizados experimentos nas bases CIFAR10, CIFAR100 e ImageNet 2012. Os resultados são satisfatórios, e o cálculo realizado em LEAP melhora o desempenho da CNN, sendo a

complexidade computacional do LEAP comparada com o desempenho igual ao de *Poolings* existentes na literatura [100].

Em outro trabalho, Z. Chen et al. [103] propõem duas abordagens (SQU e GatedSQU) que modificam as CNNs tradicionais, que por sua vez, não proporcionam bons resultados quando aplicadas em determinadas bases. Na primeira, é aplicado o cálculo de raiz quadrada nos mapas de características (*activation maps*) por canal do *Pooling* existente. SQU faz melhor uso das informações sobre os mapas e é superior ao *Average* ou *Max Pooling* no contexto de recuperação de instância (*instance retrieval*). E na segunda abordagem, faz-se uma melhoria de SQU. Cria-se um *Pooling*, em que se aprende uma função que pondera as contribuições de diferentes canais. Experimentos em seis bases de imagens mostram que as estratégias propostas obtiveram melhorias consideráveis em relação ao estado da arte.

Os autores A. M. Romano and A. A. Hernandez [104] expressam a necessidade da redução de *overthiting* em arquiteturas CNN. Assim, propõem uma modificação na camada de *Pooling* da CNN, onde é introduzido um cálculo que se baseia em rejeição, fazendo com que as características filtradas sejam encaminhadas para o restante das camadas. Experimentos foram realizados com arquitetura composta por duas camadas Convolucionais, e as bases MINIST e CIFAR-10. No uso do *Pooling* proposto, a primeira base proporcionou 98,34% de ACC, e a segunda, 98,10%. Usando *Max Pooling* obtiveram 98,10% e 91,36% respectivamente. Neste trabalho, os autores não mesclam o uso do *Pooling* proposto com outras operações, como *Max* ou *Average Pooling*, e nem em arquitetura de CNN tradicional.

4.5. Discussão

Apresenta-se na Tabela 5 o resumo dos trabalhos relacionados apresentados. Inicialmente, afirma-se que os trabalhos de M. D. More et al. [73], G. S. Simões [75] e I. Joshi et al. [76] estão próximos da realidade deste trabalho. Entretanto, estes trabalhos não são práticos. Os exemplos apresentados pelos autores são teóricos, e os experimentos apresentados foram feitos em bases de imagens que fogem do contexto diário. Por exemplo, em trabalhos que apresentam aplicações GAN, verifica-se que autores não estudam criteriosamente a GAN, e sim, a aplicação delas, como para a anonimização [70]. Por sua vez, o trabalho de M. D. More et al. [73] e G. S. Simões et al. [75] são interessantes, entretanto, fizeram a exposição apenas

de imagens de mulheres com biquíni, o que parece estranho, já que a rede foi treinada para detectar cinco classes, assim como não informam o processo [75].

Foram estudados trabalhos que visam a detecção de pornografia no uso de CNN, embora existam vários métodos para a classificação de imagem pornográfica [4]. Diversas abordagens propostas são relacionadas a vídeos. No entanto, a implementação em aplicativos do mundo real permanece um desafio em aberto. Esta questão ocorre porque, para aumentar a precisão, os pesquisadores normalmente extraem recursos em vários *frames* [8-9] ou aumentam significativamente a complexidade da CNN [10-11]. A detecção *online*, considerando a implantação real do método da proposta permanece sem abordagem na literatura. No entanto, uma abordagem de detecção de pornografia baseada em técnicas de detecção de objetos pode melhorar significativamente a experiência do usuário. Isso ocorre porque o FP ocorre apenas em uma área desfocada do vídeo, em vez de restringir o acesso a todo o vídeo. Assim, existem trabalhos limitados que usam a detecção de objetos para localizar partes íntimas, bem como a detecção de pornografia.

Os trabalhos que utilizam detectores de objeto se referem a utilização em áreas diversas, como por exemplo, aumentar significativamente a complexidade para classificação de imagens pornográficas [84]. E, tratando-se de *Pooling*, são apresentados trabalhos que envolvem abordagens para a camada de *Pooling* da CNN. As camadas de *Poolings* existentes na literatura são limitadas. Geralmente se utiliza o *Max* ou *Average Pooling*. A exemplo, o primeiro utiliza apenas os valores máximos de cada região analisada [37]. Embora seja uma área muito estudada, ainda precisa de atenção. A abordagem de R. Girshick [100] é de interesse, pois pode ser aplicada na detecção de partes íntimas. Por sua vez, os autores não deixam claro como é feita a aplicação da operação de raiz quadrada nos mapas de ativação [103]. Não é claro se os mapas de ativação são provenientes das operações de *Average* ou *Max Pooling*, ou mesmo explica o motivo de aplicar tal cálculo. Ainda, a abordagem SQU proposto em [103] não é um *Pooling*, e sim, uma abordagem, ao contrário do GatedSQU *Pooling.*

Tabela 5. Comparação entre os trabalhos relacionados.

Solução Autor Principal	**Método para publicar Base - *Bechmark***	**Impacto de FP - *user-friendly***	**Detecção de Objetos**	**CNN**	**Modificações no *Pooling* CNN**	**Censurar Imagem**	**Aprimoramento (Melhoria) em Imagens**	**Anonimizar Imagem**
T. Kim e J. Yang [70]	X						X	X
M. D. More et al. [73]						X	X	X
G. S. Simões [75]				X		X	X	
I. Joshi et al. [76]							X	
H. Hukkelås et al. [77]						X	X	X
E. Piacentino and C. Angulo [80]				X			X	X
E. Piacentino and C. Angulo [81]				X			X	X
J. Mahadeokar and G. Pesavento [9]				X				
F. Nian et al. [82]				X				
Moustafa [8]				X				
Vitorino et al. [12]				X				
M. Perez et al. [10]				X				
J. Wehrmann, G. S. Simões, R. C. Barros, and V. F. Cavalcante [11]				X				
Zhikang et al. [87]				X				
Gangwar et al. [88]				X				
X. Ou et al. [84]			X	X				
L. Wang et al. [89]			X					
G. Plastiras et al. [90]			X					
P. Maheshwari et al. [91]			X					
S. Saikia et al. [92]			X					
C. Lee et al. [94]					X			
K. He et al. [95]					X			
Z. Zhong et al. [96]					X			
M. Zeiler and R. Fergus [97]					X			

Solução **Autor Principal**	**Método para publicar Base - *Bechmark***	**Impacto de FP - *user-friendly***	**Detecção de Objetos**	**CNN**	**Modificações no *Pooling* CNN**	**Censurar Imagem**	**Aprimoramento (Melhoria) em Imagens**	**Anonimizar Imagem**
B. Graham [98]					X			
M. Jaderberg et al. [99]			X					
R. Girshick [100]			X					
R. B. Girshick [101]			X		X			
M. Sun et al. [102]			X		X			
z. Chen et al. [103]					X			

Capítulo 5

Partes Íntimas: Detecção *Online* e Ofuscação para *Benchmark* Público de Vídeo Pornográfico

Neste Capítulo apresenta-se o método, que foi dividido em duas etapas. Posteriormente, expõem-se uma discussão do Capítulo.

5.1. Visão Geral da Proposta

Na Figura 10 ilustra-se o método proposto: *uma arquitetura para detecção online de pornografia em streaming de vídeo* (PPCensor - *Private Parts Censor*) e a *publicação de imagens de pornografia sem revelar as partes íntimas das cenas* (PPOBench).

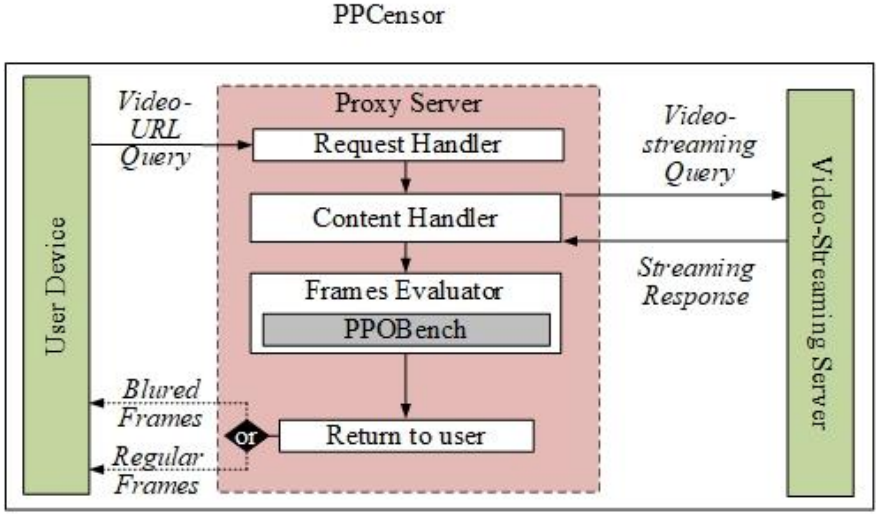


Figura 10. Método proposto. Adaptado de [21].

PPCensor aplica a técnica de detecção de objetos usando CCN para classificar e censurar *online*, apenas às cenas contendo partes íntimas num vídeo. Usando esta mesma técnica as partes íntimas são ofuscadas e disponibilizadas numa base pública, denominada de *Private Parts Object Dataset* (PPO).

O objetivo é ocultar apenas as áreas inadequadas em um conjunto de *frames*, em vez de bloquear o acesso a todo o vídeo. Consequentemente, PPCensor é *user-friendly*, visto que é possível visualizar o vídeo, embora *frames* potencialmente inadequados tenha ocultação parcial da imagem. Além disso, mesmo que ocorra um FP (falso positivo), o usuário experimentará apenas um momento (alguns *frames*) com regiões desfocadas (ofuscadas) no vídeo, em vez de bloquear incorretamente o acesso a todo o vídeo. Portanto, a experiência do usuário ao usar o PPCensor não é significativamente degradada devido à ocorrência de um FP.

Em segundo lugar, para permitir o uso real na produção, PPCensor é implementado como um servidor *proxy* para *streaming* de vídeo. O principal *insight* é avaliar de forma transparente a ocorrência de vídeo pornográfico e a classificação de *streaming* de vídeo remotamente.

Para atingir este objetivo, PPCensor é implantado em *hardware* dedicado, e realiza sua detecção de pornografia por meio de um servidor *proxy*. Consequentemente, ele avalia o conteúdo de vídeo de todas as URLs de vídeo consultadas *online*. Se for encontrado conteúdo de vídeo pornográfico, ele pode bloquear o acesso ou ocultar *frames* inadequados no servidor, classificando o conteúdo pornográfico sem a intervenção do usuário. Portanto, PPCensor requer apenas que o usuário o utilize como servidor *proxy*, possibilitando assim a filtragem de conteúdo impróprio independente do dispositivo que o usuário está usando, seja um computador ou um celular. Obviamente, existe a opção de uso de *proxy* transparente, onde nem esta configuração é necessária.

Por sua vez, PPOBench comprova que imagens com partes íntimas ofuscadas podem ser reconstituídas dentro de uma CNN e utilizadas para detecção de pornografia em casos de *benchmarks* ou retreino do modelo.

O PPOBench possui dois objetivos, (i) no primeiro formaliza-se uma base ofuscada dividida entre as classes normal e pornográfica, usando a base PPO, para realizar a reconstituição do ofuscamento de partes privadas inseridas na base. Para tal, utiliza-se uma GAN (*Generative Adversarial Network*) treinada com a PPO (gerada em PPCensor), e após a reconstituição, tem-se uma base muito similar à de treinamento do modelo de pornografia

usando CNN; (ii) no segundo, utiliza-se um classificador de imagens CNN com a camada de *Pooling* modificada e apresenta-se que é possível melhorar a reconstrução a partir da base ofuscada.

5.2. PPCensor: Uma Arquitetura para Detecção *Online* de Pornografia em *Streaming* de Vídeo

O objetivo de PPCensor é alcançar detecção *online* e com características *user-friendly*, sem a necessidade de processamento adicional no dispositivo do usuário. Para atingir esse objetivo, o PPCensor é executado em *hardware* dedicado, mas de propósito geral, e a detecção é realizada de forma transparente no dispositivo do usuário final.

PPCensor tem seu funcionamento em duas etapas principais: *Video-Streaming Query and Handling* (Consulta do *streaming* de vídeo e tratamento) e *Pornography Object Detection* (detecção de objetos pornográficos). O primeiro trata da análise de consulta do dispositivo do usuário, *download* e filtragem de vídeo, e resposta filtrada. O processo é executado inteiramente em um servidor *proxy*. Portanto, o único requisito é que o usuário final deve alterar as configurações do dispositivo para redirecionar as consultas para o servidor PPCensor. O administrador também pode impor o uso do *proxy* aplicando a técnica de *proxy* transparente, não necessitando nenhuma configuração por parte do usuário. Assim, o processo de detecção não requer processamento adicional no dispositivo do usuário e pode ser executado mesmo em dispositivos com recursos limitados para este tipo de processamento com um celular.

A etapa de detecção de objetos pornográficos é responsável por apresentar uma abordagem de detecção *user-friendly*. PPCensor é o primeiro trabalho que aborda a tarefa de detecção de pornografia como um problema de detecção de objetos. Baseia-se no *insight* de que as partes íntimas (cenas com nudez explícita) podem ser identificadas como objetos e filtradas adequadamente em vídeos, por exemplo, borrando / ofuscando a região / *frames* do vídeo.

Como resultado, mesmo que ocorra uma detecção incorreta, a experiência do usuário não é significativamente degradada. Isso ocorre porque o PPCensor trata o conteúdo impróprio como um problema de detecção de objeto e é capaz de filtrar *frames* de vídeo inadequados em vez de bloquear todo o conteúdo do vídeo.

A arquitetura de processamento do PPCensor é ilustrada na Figura 10. Essas duas etapas são descritos em detalhes nas próximas subseções, incluindo os componentes da arquitetura e a detecção de objetos aplicado à detecção do vídeo.

5.2.1. Detecção de Objetos Pornográficos

Antes de examinar os detalhes da arquitetura de processamento de PPCensor, descreve-se a abordagem de detecção de objetos pornográficos, que em contraste com os trabalhos relacionados, que detectam vídeo pornográfico por meio da classificação de *frames* do vídeo, aborda a detecção de conteúdo pornográfico como um problema de detecção de objeto.

O PPCensor é capaz de fornecer dois benefícios importantes: taxa de transferência de detecção e detecção *user-friendly*. Primeiro, a detecção de objetos na imagem pode ser alcançada quase em tempo real. Posteriormente, a filtragem de regiões específicas da imagem, em vez de toda a imagem, melhora significativamente a experiência do usuário porque, se ocorrer um FP, apenas uma parte da imagem do vídeo ficará desfocada, em oposição ao bloqueio do vídeo inteiro.

Para atingir esse objetivo, PPCensor considera cada parte íntima, feminina e masculina, como um objeto. Assim, o objetivo do PPCensor é detectar quatro classes de objetos específicas: pênis, vaginas, seios e nádegas.

É importante observar que essas partes íntimas não são comumente compartilhadas em vídeos regulares (sem pornografia); em contraste, elas costumam estar presentes em vídeos pornográficos. Por exemplo, em um único *frame*, um filme explicitamente pornográfico pode mostrar todas as partes íntimas, dependendo apenas da posição da câmera e do objetivo das cenas.

Portanto, ao considerar as partes íntimas como objetos, PPCensor é capaz de detectar pornografia em vídeos usando técnicas de detecção de objetos baseada em CNN. Por sua vez, a detecção de objetos se torna o método semiautomático para associação de tarja. Ou seja, cria-se um ofuscamento padronizado, importante para a segunda etapa do método proposto, onde formaliza-se a base Pt.

5.2.2. Arquitetura de Processamento

O objetivo do PPCensor é realizar a detecção de objetos com base em conteúdo pornográfico *online*, sem qualquer processamento adicional no dispositivo dos usuários. Nesse sentido, PPCensor é implementado como um servidor *proxy* e a arquitetura de processamento é ilustrada na Figura 10.

A arquitetura de processamento considera o conjunto de *user-device* (dispositivo do usuário) que precisa avaliar e filtrar sua mídia de vídeo. O usuário deve apenas configurar seu dispositivo para encaminhar qualquer consulta ao servidor *proxy* PPCensor, utilizando-o como *proxy* de rede. Deste modo, todas as consultas de *streaming* de vídeo realizadas no dispositivo do usuário são encaminhadas para o PPCensor. Por sua vez, ele recebe as consultas realizadas por meio do módulo *content handler* (consulta), que é responsável pelo recebimento e análise de consultas de URLs. O objetivo do módulo é estabelecer uma consulta de vídeo em URL para processamento posterior.

Na sequência, o módulo manipulador de conteúdo consulta o servidor de *streaming* de vídeo para obtenção dos *frames* do vídeo através do servidor *proxy*, e os encaminha para o módulo *frames evaluator* (avaliador de *frames*) que executa o mecanismo proposto de detecção de objetos pornográficos e rotula apropriadamente todos os objetos detectados no *frame*.

Finalmente, o módulo filtro de conteúdo realiza a filtragem *online* dos *frames* recebidos para o dispositivo do usuário. Por exemplo, desfocando (ofuscando) os objetos pornográficos detectados ou até mesmo bloqueando o vídeo, se este for o desejo do usuário.

5.3. PPOBench: Publicação de Imagens de Pornografia sem Revelar as Partes Íntimas das Cenas

Por sua vez, o PPOBench representa um método para publicação de imagens de pornografia sem revelar as partes íntimas das cenas. Ele comprova que imagens ofuscadas (i.e., partes íntimas tarjadas) podem ser utilizadas para detectar pornografia. O PPOBench faz a reconstituição das partes ofuscadas da PPO usando uma GAN e uma CCN com a camada de *Pooling* modificada para permitir o *benchmarking* sem expor publicamente as partes íntimas na base. O funcionamento do mecanismo, dividido em 2 etapas é representado por meio da arquitetura de processamento do PPOBench, conforme visto na Figura 11.

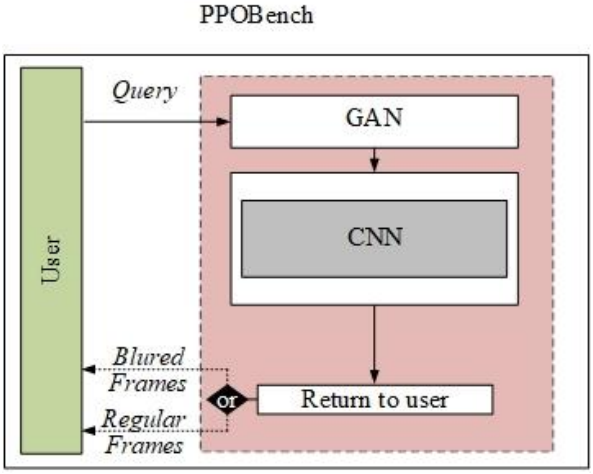


Figura 11. Arquitetura de processamento do PPOBench.

5.3.1. GAN - Reconstituição

Objetiva-se reconstituir o ofuscamento inserido em imagens de pornografia mediante a aplicação de GAN, sendo que o processo visa diminuir o impacto no treinamento e classificação da CNN, sem deixar os resultados da reconstituição acessíveis. O processo ocorre em duas etapas, (i) na primeira, treina-se uma GAN onde gera-se um modelo, sendo que para o treinamento, são utilizadas aleatoriamente objetos (imagens) pornográficas. Considera-se como objetos, imagens com partes íntimas divididas entre as classes: pênis, vagina, nádegas e seios.

O modelo gerado pela GAN realiza a reconstituição das imagens pornográficas ofuscadas. As imagens são submetidas ao modelo GAN individualmente. O modelo tem como entrada a imagem ofuscada (produzida por PPCensor), e como saída, a imagem reconstituída. Gera-se uma base reconstituída. Por fim, como a GAN realiza modificações na distribuição dos *pixels* da imagem de saída, todas as imagens normais e as imagens de teste pornográficas, que não estão ofuscadas, devem ser submetidas à GAN, formalizando a base que será submetida a próxima etapa.

5.3.2. CNN - Detecção de Imagem Pornográfica

A etapa da CNN gera modelo de detecção de imagem de pornografia a partir das imagens reconstituídas pela GAN. Fraciona-se a base reconstituída em uma razão, por exemplo de 60/20/20 (treinamento/validação/teste, respectivamente), sendo que as imagens de treinamento e validação são submetidas ao algoritmo de CNN, com o intuito de obtenção do modelo. Após geração do modelo, é realizada a classificação, ou seja, o uso-detecção do modelo gerado no uso das imagens de teste. Para validação do uso-detecção é analisada a precisão (ACC) considerando-se o grau de confiança, por exemplo, de 50%.

Optou-se pela utilização da confiança pelo fato da última camada do modelo CNN gerar a probabilidade de uma imagem pertencer a determinada classe. Nos trabalhos relacionados isto não é comentado. Imagina-se que os autores usam grau de confiança implícito de 50% em sua maioria, dado que são duas classes.

Para minimizar o efeito das imagens reconstituídas durante a obtenção do modelo e do uso-detecção, realiza-se a intervenção na camada de *Pooling* da CNN. Na prática, quando é feita a reconstituição, o conteúdo ofuscado na imagem é parcialmente reconstituído pela GAN, i.e., é possível a visualização de falhas (“buracos”) em partes do antigo ofuscamento. Por isso, diminui-se os efeitos que a GAN gera nas imagens reconstituídas durante a obtenção do modelo e seu uso-detecção com a CNN. Assim, é possível minimizar o efeito dos *pixels* comprometedores (indesejados), assim como [103].

Uma arquitetura CNN tradicional pode possuir várias camadas de *Pooling* [33], sendo que normalmente essa camada executa operações tradicionais (*Average* ou *Max Pooling* [5]). Por exemplo, a arquitetura CaffeNet [68] possui três camadas de *Pooling*, e neste caso, por três vezes realiza-se a operação de *Max Pooling*.

Deste modo, conforme exemplificado na Figura 12, que apresenta um exemplo de funcionamento do método proposto é feita a aplicação da operação matemática da raiz quadrada no mapa de características resultante do *Max* ou *Average Pooling*. Assim, existe a inserção/modificação da operação realizada no *Pooling* [103]. Diferente do trabalho em que apenas se aplica a operação de raiz quadrada no mapa de características do *Pooling.* Ademais, uma CNN proporciona bons resultados no treinamento e classificação para algumas bases de imagens quando todas as camadas de *Pooling* tem aplicação da operação de raiz quadrada no mapa de características.

Em outras palavras, no método proposto é aplicado em um ou mais *Poolings*, a função da raiz quadrada após a execução do *Max* ou *Average Pooling*. Esta abordagem tende a diminuir o valor de cada *pixel* da imagem, fazendo com que os *pixels* não relevantes sejam descartados ao longo da execução/processamento da CNN. A inserção/modificação da operação realizada pode ser feita na combinação de *Poolings*, e não em todos os *Poolings* da CNN. Faz-se um aprimoramento (não é feito correção) nas imagens (treinamento, validação e teste) durante a obtenção e uso-detecção do modelo.

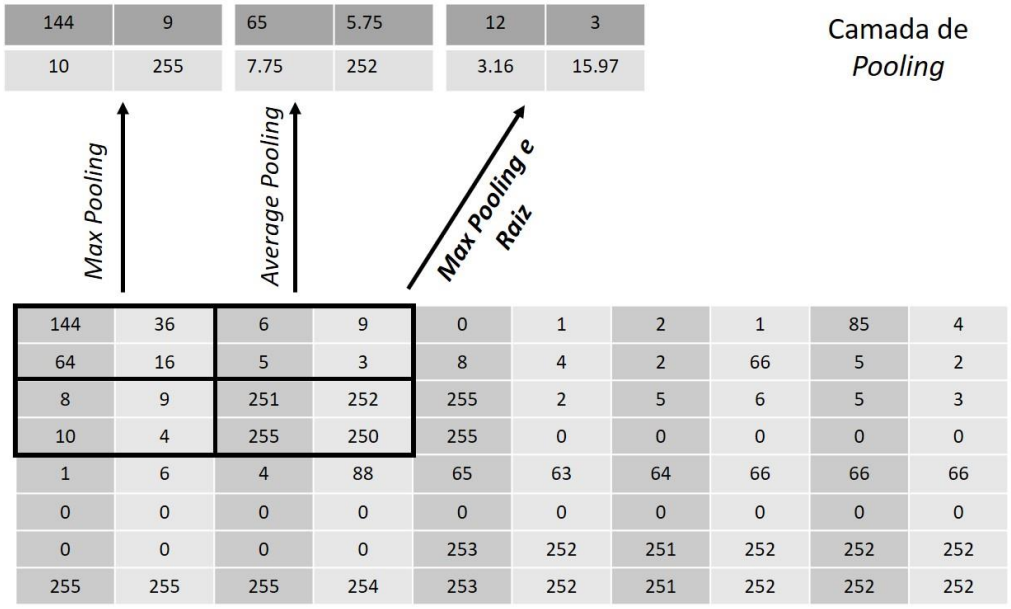


Figura 12. Exemplo de funcionamento.

Como análise dos resultados, compara-se visualmente através da técnica *Explainable* [42] e verifica-se valores da matriz de confusão e respectivas ACCs provenientes do uso-detecção do modelo treinado para detecção de pornografia.

5.3.3. Discussão

O método proposto está dividido em duas etapas: PPCensor e PPOBench, sendo que os resultados da aplicação dos métodos são apresentados no Capítulo 6. A primeira etapa diferencia-se da literatura, já que ela não considera o processamento (recursos computacionais) exigido pelo esquema de detecção (PPCensor considera os dispositivos com recursos limitados).

Para tal, propõem-se uma arquitetura para detecção *online* de conteúdo pornográfico em vídeos que é baseada em dois *insights* principais: é a primeira abordagem que trata da detecção de conteúdo pornográfico como um problema de detecção de objetos. Portanto, PPCensor é capaz de melhorar a experiência do usuário, considerando que um FP não bloqueia o acesso do usuário ao vídeo consultado, mas apenas ofusca regiões específicas do *frame*, tornando a experiência do usuário com características *user-friendly*.

Em segundo lugar, é fornecido detecção de conteúdo pornográfico *online*, sem qualquer processamento adicional do dispositivo do usuário. Salienta-se que PPCensor é implementado como um *proxy* de rede. Consequentemente, o vídeo acessado é avaliado de forma transparente e remota, sem a necessidade de interação do usuário.

Na segunda etapa é deflagrado PPOBench, que ao contrário dos trabalhos disponíveis na literatura, provê uma base de pornografia sem revelar as partes íntimas nas imagens pornográficas. O método faz a reconstituição das partes ofuscadas da PPO. Mediante aplicação de PPOBech comprova-se que imagens ofuscadas podem ser usadas para a detecção de pornografia, e sendo assim, utilizada por outros autores na realização de *benchmark*, o que atualmente não é possível.

Capítulo 6

Avaliação

Este Capítulo apresenta os resultados de experimentos aplicados no uso de PPCensor e PPOBench.

6.1. PPCensor

Na atual seção, apresenta-se a base PPO e a prototipação de PPCensor. Por fim, responde-se as seguintes questões de pesquisa: R1: É possível tratar a detecção de vídeo pornográfico como uma tarefa de detecção de objeto? R2: Quão desafiadora é a detecção de cada parte íntima? R3: Como o PPCensor executa as tarefas de classificação de imagem em comparação com CNNs de última geração? R4: O PPCensor facilita a detecção *online* de *streaming* de vídeo pornográfico?

6.1.1. PPO *(Private Parts Object Dataset)*

Para a execução de PPCensor é formalizada a base PPO (*Private Parts Object Dataset*) [21]. Como aborda-se a detecção de pornografia como um problema de detecção de objeto, são processadas apenas as imagens da classe pornográfica da base Pornography-2k [66].

A tarefa de pré-processamento filtra as imagens resultantes da extração dos vídeos de acordo com seu conteúdo e apenas as imagens explicitamente pornográficas foram selecionadas. A partir desse subconjunto, foi selecionado manualmente em cada imagem, as regiões contendo os objetos como pênis, vagina, seios e nádegas.

Para atingir o objetivo, é definido uma região de caixa delimitadora que contém cada objeto pornográfico considerado. Como resultado, o conjunto de dados final consiste em, por exemplo, 50.870 imagens pornográficas, das quais são selecionados, por exemplo, 52.215 objetos que contêm 13.607 pênis, 11.346 vaginas, 13.963 seios e 13.299 nádegas. Entre os exemplos de objetos, uma única imagem pode conter mais de um objeto. Na Figura 13 são apresentados quatro exemplos de objetos. Observa-se que uma única imagem pode conter mais de um objeto.

As imagens da PPO foram divididas em três partes: treinamento, validação e teste (60/20/20). Consequentemente, cada parte é composta por vídeos exclusivos, cada um com objetos pornográficos exclusivos, avaliando adequadamente a capacidade de generalização do método.

Figura 13. Exemplos de imagens da base PPO.

A base PPO foi utilizada para gerar modelo que detecta objeto pornográfico (explanação na próxima subseção).

6.1.2. Protótipo

O protótipo para PPCensor é implementado e implantado em um processo *multithread*, conforme mostrado na Figura 14, sendo que o protótipo considera três entidades principais: *User Device*, *Proxy Server* (implementação de PPCensor), e *Video-Streaming Server*.

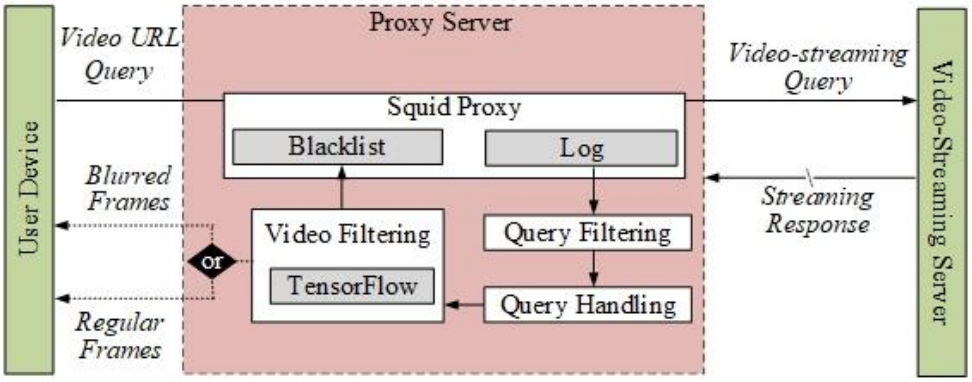


Figura 14. Arquitetura de implementação do protótipo PPCensor [21].

O processo *Video-URL Query* é responsável pela consulta de vídeo no PPCensor. Nas configurações de implantação de produção, ele é representado pela requisição do dispositivo do usuário (*user device*).

Para este fim, um programa em linguagem Python é implementado para realizar vários *Video-URL Queries*. Todas as consultas realizadas são passadas para o Servidor PPCensor (*proxy*) que foi implementado usando o Squid *proxy*, uma ferramenta de servidor *proxy* com suporte da web. O nó *Video-URL Query* é configurado para usar o PPCensor como seu servidor *proxy*. Como resultado, não há alterações adicionais no dispositivo do usuário final.

O servidor PPCensor é implementado usando três processos principais. Primeiro, a implementação do *proxy* é executada usando o *Squid Proxy* [105] versão 4.5. Esta ferramenta está publicamente disponível e é amplamente utilizada para *proxy* de rede em servidores baseados em Linux. No decorrer do tempo, o *Squid Proxy* registra todas as consultas recebidas em um arquivo de log, que é então analisado e interpretado por outro processo, o *Query Filtering and Handling* (Consulta de Filtragem e Tratamento).

Esses processos são módulos que são implementados em Python e filtram continuamente o log do *Squid Proxy*. Quando uma URL de vídeo é encontrada, a mídia correspondente é baixada do *Video-Streaming Server* pela API Pafy [106] versão 0.5.1. O processo executa o *download* do vídeo em 0,5 *frames*/segundo e um único *frame* é baixado para cada intervalo de vídeo de 2 segundos, que é temporariamente armazenado (em *cache*) em uma pasta para processamento posterior.

Finalmente, a abordagem de detecção de objetos de pornografia proposta é implementada usando Python em outro processo. O referido executa a API do TensorFlow *Detect Object* [106] usando tensorflow-gpu versão 1.14.

Assim, o módulo é executado continuamente, e quando um novo *frame* de vídeo é baixado, o modelo CNN é aplicado e os objetos pornográficos identificados são rotulados. O módulo também permite a inserção de *black list* de vídeos por meio do recurso de lista negra do *Squid Proxy*. Portanto, um vídeo classificado de pornografia também pode ser bloqueado em tempo real, se necessário.

Por fim, o *Video-streaming Server* objetiva responder às *Video-URL Query* e *Video-Streaming Queries*. É importante observar que se a *black list* do servidor PPCensor encontrar uma URL pornográfica, o *download* da URL na *black list* será interrompida imediatamente e o usuário final não é capaz de acessar o vídeo, mesmo se o *streaming* de vídeo estiver atualmente em *download*.

6.1.3. Acurácia de PPCensor

Sobre a acurácia de PPCensor, para avaliar a viabilidade da detecção de objetos em PPCensor, primeiro aborda-se a questão se é possível tratar a detecção de vídeo pornográfico como uma tarefa de detecção de objeto. Para atingir este objetivo, são avaliadas várias arquiteturas de CNN para a detecção de objeto no uso da base PPO. Cada arquitetura CNN é executada com, por exemplo, 400.000 *steps*. *Learning rate* e *decay rates* são definidas empiricamente para cada arquitetura após vários experimentos. Além disso, todas as arquiteturas CNN avaliadas são treinadas usando *transfer learning* [49-50] utilizando os pesos obtidos em uma base de imagens COCO [51], uma base amplamente utilizada para fins de detecção de objetos.

As técnicas baseadas em CNN são avaliadas em relação ao seu rendimento de detecção e mAP. O mAP é calculado de acordo com uma interseção sobre união (IoU), aplicando um limite de 0,5 para cada caixa delimitadora de objeto. A Tabela 6, de avaliação do desempenho do PPCensor na detecção de objetos pornográficos, mostra o desempenho das melhores arquiteturas de detecção de objeto baseada em CNN aplicadas.

Tabela 6. Avaliação do desempenho do PPCensor na detecção de objetos pornográficos. Adaptado de [21].

Arquitetura CNN	*Frames*/sec	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

Pode-se observar que existem diferenças significativas entre os valores de ACC obtidos por cada arquitetura CNN avaliada. Em relação a CNN mais precisa, *Faster R-CNN Inception* obteve um mAP de 63,50%. É importante notar que o mAP considera toda a interseção do objeto identificado de pelo menos 50%, nesta avaliação. No entanto, em termos de uso do ambiente de produção, a região da caixa delimitadora identificada pode ser aumentada (i.e., aumentando a região desfocada) para melhorar a ACC do modelo sem compensação significativa da experiência do usuário. Consequentemente, a arquitetura mais precisa foi de 63,50% em mAP para o *Faster R-CNN Inception*, o que mostra a viabilidade de tratar o *streaming* de vídeo pornográfico através da detecção de objeto.

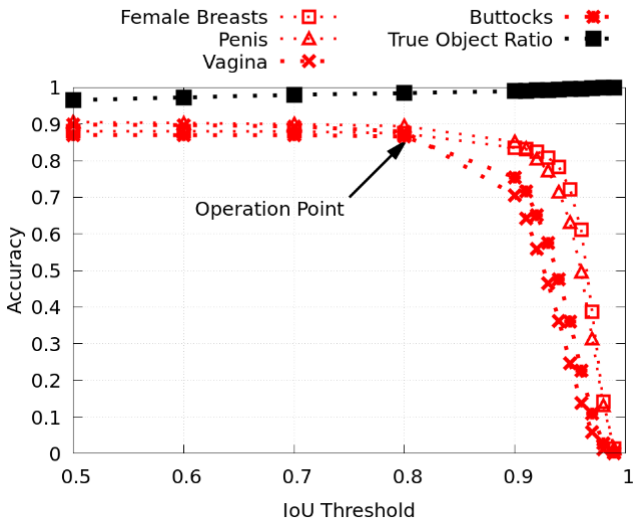


Figura 15. *Faster R-CNN Inception* IoU e ponto de operação. Adaptado de [21].

Investigou-se ainda mais a relação entre o limite de IoU e as taxas de ACC, conforme mostrado na Figura 15, que apresenta um gráfico do *Faster R-CNN Inception* IoU e ponto de operação. Pode-se observar que o limite de IoU é capaz de melhorar a relação do objeto detectado com uma pequena perda para cada taxa de detecção de objeto até o limite de 0,8 IoU.

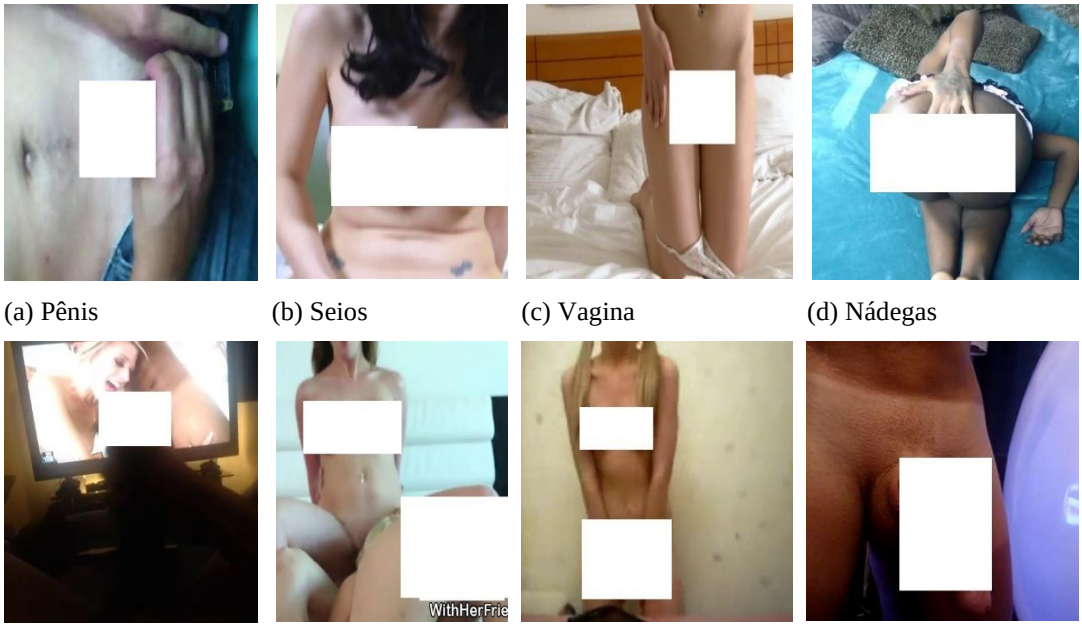
Enfim, para saber quão desafiadora é a detecção de cada parte íntima, inspecionou-se ainda mais a taxa de detecção de cada parte íntima. Para atingir esse objetivo, primeiro é avaliado o *Faster R-CNN Inception* com um IoU de 0,8 (referente ao ponto de operação da Figura 15) e a matriz de confusão, conforme mostrado na Tabela 7.

Tabela 7. *Faster R-CNN Inception* (IoU 0.8) - matriz de confusão. Adaptado de [21].

Classe – Objeto (Parte Íntima)	Classificado como				
	Falso Negativos	Seios	Pênis	Vagina	Nádegas
Seios	271	2186	38	0	5
Pênis	247	2	2236	11	4
Vagina	258	0	4	2199	39
Nádegas	285	5	7	38	2165

Pode-se observar que, em geral, as partes íntimas não são classificadas incorretamente, mas sim, não identificadas como um objeto (conforme Falsos Negativos na Tabela 7). Isso significa que a CNN é capaz de identificar que tipo de parte íntima foi encontrada. Ressalta-se que dois objetos apresentam maior ocorrência de FP: seios e pênis.

Por meio de exemplos de imagens pornográficas rotuladas com a arquitetura *Faster R-CNN Inception*, é possível notar como a abordagem de detecção de objetos proposta impacta a experiência do usuário (Figura 16). As regiões nas imagens com objetos de parte íntima são devidamente identificadas e filtradas e a experiência do usuário não é significativamente degradada. Isso ocorre porque o usuário ainda é capaz de identificar o contexto da cena sem ser exposto à nudez explícita. Exemplos comuns de imagens normais com objetos detectados incorretamente (FP), quando a arquitetura *Faster R-CNN Inception* é aplicada nas imagens de uma base UCF101 (Figura 17), incluem um braço ou dedo identificado erroneamente como pênis e regiões arredondadas sendo erroneamente rotuladas como seios. Por meio desses exemplos, é possível notar que a identificação incorreta de um objeto não apresenta um grande impacto na experiência do usuário. Isso significa que a experiência do usuário final não é relativamente degradada quando ocorre um FP. Isso ocorre porque apenas uma pequena parte do vídeo é filtrada ou desfocada.

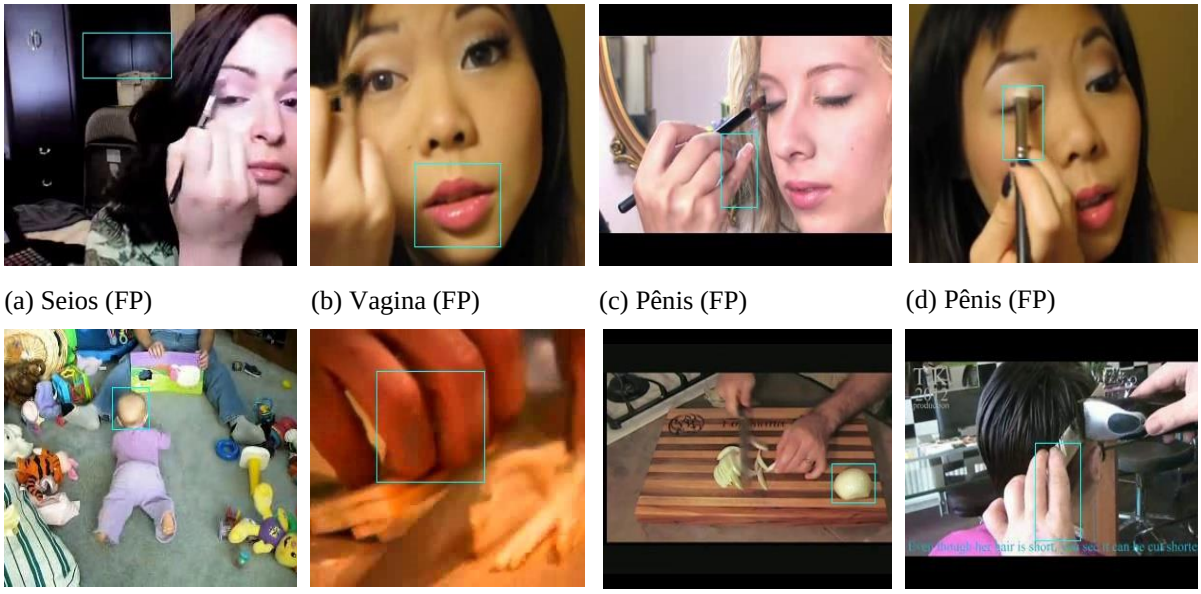


(a) Pênis (b) Seios (c) Vagina (d) Nádegas

(e) Pênis (f) Seios e Nádegas (g) Seios e Vagina (h) Pênis

Figura 16. *Faster R-CNN Inception* (IoU 0.8) - exemplos de detecção das partes íntimas realizada por PPCensor. Adaptado de [21].

Finalmente, em algumas circunstâncias, o administrador do sistema pode preferir bloquear o vídeo inteiro, ou partes dele, em vez de filtrar regiões do *frame* de vídeo específicas conforme executado pelo PPCensor. Por exemplo, no caso de filmes explicitamente pornográficos, é preferível bloquear o acesso ao filme inteiro em vez de permitir o acesso a um vídeo filtrado. Portanto, para abordar a questão como o PPCensor executa as tarefas de classificação de imagem em comparação com CNNs de última geração (R3), também é avaliado o PPCensor durante a aplicação ao campo de classificação de imagens. Para atingir esse objetivo, o PPCensor foi customizado para classificar um *frame* de vídeo como pornográfico se algum objeto íntimo fosse identificado no *frame* de vídeo. Os resultados do PPCensor também foram comparados com as arquiteturas CNN de última geração usadas para tarefas de classificação de imagens.



(a) Seios (FP) (b) Vagina (FP) (c) Pênis (FP) (d) Pênis (FP)

(e) Seios (FP) (f) Nádegas (FP) (g) Seios (FP) (h) Pênis (FP)

Figura 17. *Faster R-CNN Inception* (IoU 0.8) - exemplos comuns de FP: UCF101. Adaptado de [21].

É utilizado um subconjunto de vídeos não utilizados da base Pornography-2k. Os vídeos são extraídos e divididos em treinamento, teste e validação, por exemplo de 60/20/20, respectivamente. Por fim, CNNs realizam a obtenção de modelo, tendo o objetivo de classificar imagem pornográfica. O treinamento é realizado usando *transfer learning* [49-50], e os experimentos executados em diversas arquiteturas de CNN, tais como CaffeNet [68], AlexNet [69] e Inception [56], com parâmetros tais como: 400.000 *steps*, *learning rate* e *decay rate* individualmente definidos empiricamente.

Os modelos obtidos foram avaliados nas imagens de subconjunto teste da base Pornography-2k e da base UCF101. Para a segunda base, qualquer imagem de vídeo classificada como pornográfico foi considerado um FP, visto que o conjunto de dados consiste apenas em imagens normais.

A Tabela 8 apresenta uma comparação do desempenho do PPCensor em comparação com as técnicas tradicionais de classificação de imagem para os vídeos da base Pornography-2k e UCF101. Surpreendentemente, o PPCensor supera as taxas de FP de todas as técnicas de última geração avaliadas na base Pornography-2k, ao mesmo tempo em que supera o desempenho de todos, exceto o Yahoo Detector no conjunto de dados da UCF101. Em relação às taxas de FN, o PPCensor também apresenta resultados semelhantes às outras técnicas, atingindo 2,34% para a base Pornography-2k, um aumento de apenas 0,31% da taxa de FN para a melhor arquitetura CNN.

Tabela 8. Comparação do PPCensor quando aplicado a tarefas de classificação de imagens. Adaptado de [21].

Técnica de Detecção	Detecção *Throughput* (*Frames*/sec)	Base		
		Pornography-2k (teste)		UCF101
		FP (%)	FN (%)	FP (%)
Yahoo Detector [9]	3,67	12,80	5,53	1,61
NuDetective [65]	411,64	58,48	3,06	23,25
CaffeNet [68]	63,97	3,41	3,70	9,68
Alexnet [69]	61,12	5,08	4,61	13,69
Inception [50]	18,29	2,46	2,03	8,03
PPCensor (*Faster R-CNN Inception*)	10,87	1,66	2,34	3,41

O *throughput* de detecção do PPCensor é significativamente pior em comparação com as outras arquiteturas CNN avaliadas. Pode-se melhorar o *throughput* da técnica proposta aplicando as outras arquiteturas CNN de detecção de objetos avaliadas (Tabela 8). Por exemplo, uma taxa de 1,19 *frames* / segundo foi obtida com apenas uma compensação de 56,53 mAP.

Além disso, a avaliação do *frames* é apenas uma parte do PPCensor (Figura 14). Apesar da aplicação de uma CNN com um alto rendimento de classificação, o restante do processo também deve ser aprimorado, mais especificamente, *downloads* e filtros de vídeo *online*.

Na próxima subseção, verifica-se a performance e a escabilidade relativos a conexão para tarefas de classificação de vídeo *online*.

6.1.4. Performance e Escalabilidade

Verifica-se ainda a performance e a estabilidade relativos à conexão para tarefas de classificação de vídeo *online*. No sentido de verificar se o PPCensor facilita a detecção *online* de *streaming* de vídeo pornográfico, é avaliado se o PPCensor facilita a detecção transparente e *online* de conteúdo pornográfico sem qualquer processamento adicional do dispositivo monitorado usando um servidor *proxy*. São utilizados dois nós: o dispositivo do usuário e o PPCensor. Ambos os nós são equipados com CPU Intel I7 (quad-core) e 16 GB de memória. Além disso, o nó PPCensor também é equipado com uma GPU Nvidia Titan-XP, que executa a arquitetura *Faster R-CNN Inception* para tarefas de detecção de objetos, com um limite de mAP de 0,8.

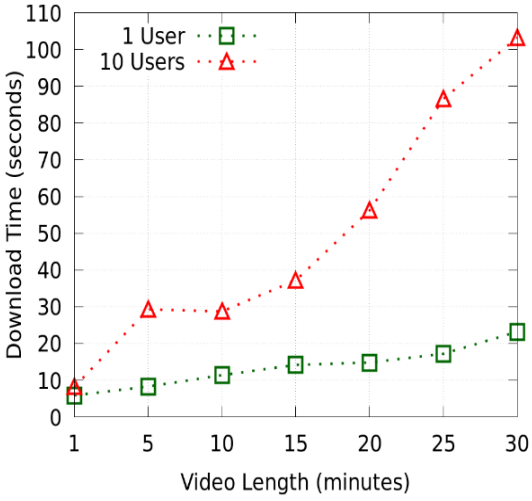


Figura 18. Tempo para PPCensor fazer o *download* do vídeo. Adaptado de [21].

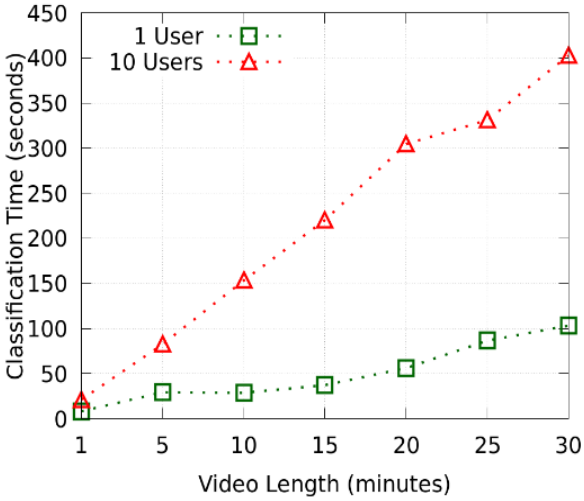


Figura 19. Tempo para PPCensor fazer a classificação. Adaptado de [21].

A Figura 18, que apresenta um gráfico de tempo para PPCensor fazer o *download* do vídeo, mostra a relação entre o tamanho do vídeo e o tempo de *download* de acordo com o número de conexões. Observa-se que o PPCensor é capaz de realizar *downloads* de vídeos *online* para até 10 conexões de usuários, sendo que o tempo de *download* é sempre menor que a duração do vídeo. Além disso, o tempo de *download* aumenta linearmente para um cenário de 1 usuário, enquanto há um aumento significativo ao executá-lo para 10 usuários. Isso ocorre porque o *download* do vídeo é vinculado à rede e à CPU. Consequentemente, dado que o nó avaliado tem apenas 8 *threads*, o tempo de execução diminui significativamente após um limite de duração de vídeo específico (20 minutos).

A Figura 19, que apresenta o tempo para PPCensor fazer a classificação, mostra a compensação entre o tamanho do vídeo e o tempo de classificação. Pode-se observar que o tempo de execução aumenta quase linearmente para cenários de 1 e 10 usuários. Esse comportamento ocorre principalmente porque a tarefa de classificação é vinculada à GPU. Portanto, a avaliação dos *frames* de vídeo é realizada individualmente, ao invés de simultaneamente. Apenas um modelo é carregado na memória para a implementação do protótipo.

A Figura 20, que apresenta a compensação entre o tempo e o tamanho do vídeo para todo o processo PPCensor para *download* de vídeo e classificação. Da mesma forma, a detecção pode ser realizada *online* para cenários de 1 e 10 usuários. Isso ocorre porque o tempo de execução do processo permanece inferior à duração do vídeo. Além disso, o tempo de

processamento necessário não aumenta significativamente em comparação com o tempo de classificação. Isso ocorre devido à natureza vinculada à GPU da tarefa de classificação e à natureza vinculada à CPU do processo de *download*. Portanto, cada processo, classificação e *download* não se degradam. Como resultado, o PPCensor permite a detecção de vídeo pornográfico quase em tempo real, sem qualquer processamento adicional no dispositivo do usuário.

Finalmente, a Figura 21, que apresenta o tempo de processamento do PPCensor para vídeos de 5 minutos e de alta qualidade, mostra a avaliação em relação ao número de usuários e o tempo de execução de vídeos de 5 minutos. Nesse caso, o PPCensor pode acomodar até 35 usuários assistindo a vídeos *online* com qualidade padrão e 25 usuários assistindo a vídeos de alta qualidade. Como tal, PPCensor detecta transparentemente *streaming online* de vídeo pornográfico para até 35 usuários sem incorrer em qualquer processamento adicional do dispositivo de uma maneira *user-friendly*. É importante observar que a implementação do protótipo pode ser ajustada para melhorar vários detalhes, aumentando ainda mais o *throughput* de detecção.

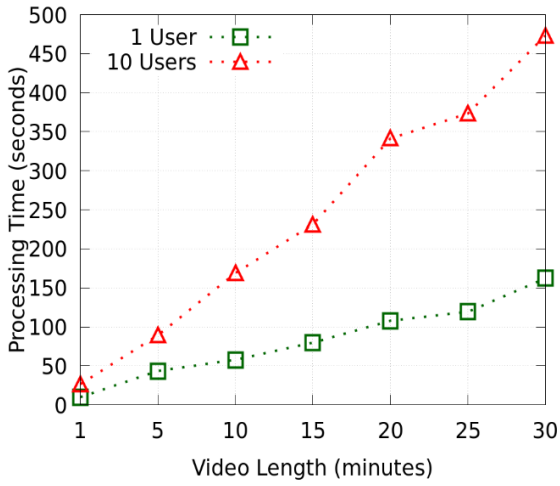


Figura 20. Tempo para PPCensor fazer o *download* e classificação do vídeo. Adaptado de [21].

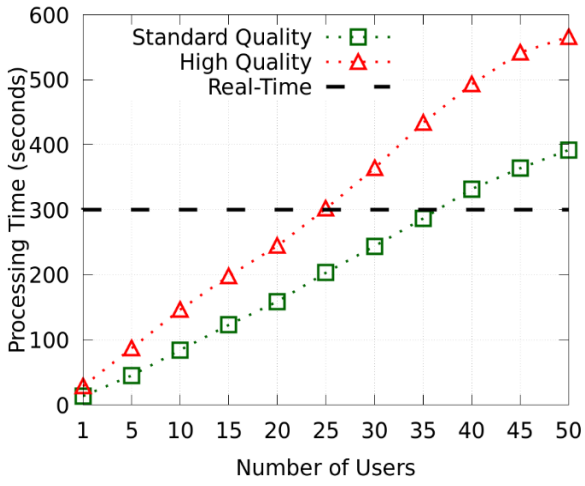


Figura 21. Tempo de processamento do PPCensor para vídeos de 5 minutos e de alta qualidade. Adaptado de [21].

6.1.5. Discussão

As ferramentas atuais publicamente disponíveis para a detecção de conteúdo pornográfico em *streaming* de vídeo não facilitam uma detecção *user-friendly* e transparente. PPCensor se baseia em duas percepções principais com relação à detecção de objetos (CNN) pornográficos.

PPCensor é o primeiro trabalho que trata o conteúdo pornográfico como um problema de detecção de objetos. Consequentemente, ele fornece detecção *user-friendly* sem impacto significativo quando ocorre FP.

A avaliação baseada na análise manual de mais de 50 mil imagens de partes íntimas revelou que a técnica proposta pode detectar conteúdos inadequados em vídeo *online*. Além disso, gera resultados semelhantes a outros trabalhos acadêmicos para tarefas de classificação de imagens. Ademais, existe a possibilidade de aumentar ou melhorar (pré-processar) os objetos rotulados.

Em segundo lugar, a arquitetura PPCensor é implementada como um servidor *proxy*; portanto, não incorre em processamento adicional no dispositivo do usuário com execução transparente. A avaliação realizada para um *desktop* com uma configuração típica mostra que o PPCensor é capaz de acomodar até 35 conexões *online* de usuários simultaneamente.

6.2. PPOBench

Apresentam-se ainda resultados obtidos com PPOBench, formalizadas as bases Pu e Pt, protótipo e a geração de modelo CNN para classificação de imagem pornográfica. Por fim, responde-se à questão da possibilidade de aplicação de GAN para reconstituição de uma base ofuscada e mesmo assim detectar pornografia.

6.2.1. Bases Formalizadas – Pu e Pt

Para execução de PPOBench são formalizadas duas bases de imagens a partir da base Pornography-2k [66]. Entre outros, tem-se se o objetivo de usar as bases para gerar modelo CNN que detecte pornografia.

A primeira base foi denominada de Pu (pornografia sem ofuscamento) produzida a partir da seleção (pré-processamento) dos *frames* da Pornography-2k. I.e., Pu possui imagens pornográficas e normais. Pu é dividida em três partes: treinamento, validação e teste. Cada parte é composta por 60/20/20 das imagens, ou dos vídeos respectivamente. Por fim, a quantia de imagens da base Pu é, por exemplo: treinamento (245.478), validação (81.826) e teste (81.826).

A segunda base foi denominada de Pt (ela possui as imagens pornográficas das porções de treinamento e validação ofuscadas). O ofuscamento está relacionado a fixação de censura que não identifique as partes íntimas através da aplicação do método semiautomático [21] nas imagens do treinamento e validação de Pu.

Por fim, Pt tem as mesmas imagens que Pu, entretanto, tem, por exemplo, 122.739 (treinamento) e 40.913 (validação) imagens ofuscadas. Salienta-se que as imagens de teste de Pu e Pt são iguais, e não possuem ofuscamento.

Figura 22. Exemplos de imagens pornográficas (treinamento) – Pu (linha 1) e Pt (linha 2). Na prática, essas imagens não estão ofuscadas. Elas foram censuradas para efeitos de edição.

Exemplos de imagens da porção do treinamento são apresentados na Figura 22. Na primeira linha são apresentadas imagens pornográficas de Pu, e na segunda linha, imagens pornográficas de Pt. Todas as imagens normais não possuem ofuscação, assim como as imagens de teste pornográficas das duas bases (Figura 23).

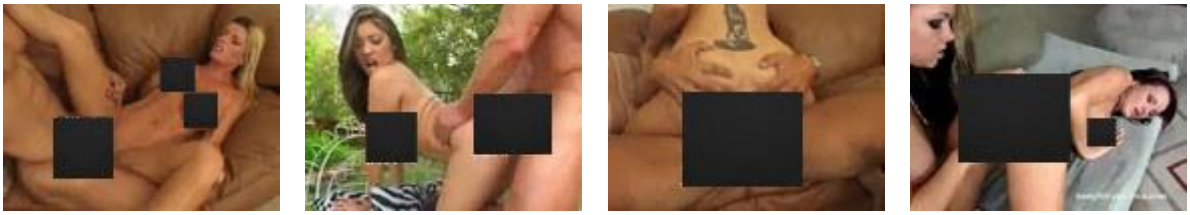


Figura 23. Exemplos de imagens pornográficas (teste) – Pu e Pt. Na prática, essas imagens não estão ofuscadas. Elas foram censuradas para efeitos de edição.

Na Figura 24 apresentam-se quatro exemplos de imagens normais que compõem o teste, tanto de Pu como Pt.

Figura 24. Exemplos de imagens não pornográficas (teste) – Pu e Pt.

A porção das bases destinadas ao treinamento são usadas para fins de treinamento. A validação para o processo de ajuste de treinamento do modelo e o teste para a avaliação final do modelo.

6.2.2. Protótipo

Explica-se a prototipação de PPOBench em três etapas. A primeira etapa foi denominada de *GAN Treinada*. Nesta é feita a reconstituição da imagem que o usuário informa no momento do teste. Aplica-se na imagem o modelo gerado por uma GAN desenvolvido em linguagem Python.

A pix2pixHD [54] é treinada e escolhida em virtude dos ótimos resultados obtidos durante os experimentos realizados com PPOBench. Para seu treinamento, são utilizados 4.000 objetos da base PPO [21], sendo que a escolha dos objetos é aleatória, entretanto divididos igualmente entre as classes da PPO. Os parâmetros utilizados no treinamento e teste da pix2pixHD foram os *defaults*, exceto *resize_or_crop* que foi definido com o valor *scale_width_and_crop* e resolução 512K, por limitações de GPU. Nos treinos, optou-se na

geração de modelo com o número de épocas igual a 250. Este modelo é responsável pela reconstituição das imagens de entrada.

Na segunda etapa de PPOBench faz-se a *geração de modelo CNN* para a detecção de pornografia. Para tal, foi utilizado o *framework* Caffe[20], visto que este permite alterações nas camadas da CNN, as quais estão implementadas em linguagem C++. Tal procedimento foi realizado na arquitetura CaffeNet [68].

CaffeNet é composta por várias camadas, sendo três *Poolings* que executam a operação de *Max Pooling*, que neste caso, posteriormente, terá a inserção/modificação de operação realizada no mapa de características (raiz quadrada) do *Pooling*. Deste modo, é denominado cada *Pooling* da CaffeNet como sendo M (*Max Pooling*). Assim, CaffeNet possui MMM (três camadas *Pooling* que executam a operação *Max Pooling*).

Neste caso, a implementação da operação de raiz quadrada no *Pooling* se consistiu em realizar a operação de *Max Pooling*, e na sequência aplicar a raiz quadrada no mapa de características resultante. Os parâmetros para utilização deste *Pooling* são os *defaults*, considerado *stride* e *kernel_size* que podem ser configurados de acordo com o valor desejado. Para treinamento da arquitetura CaffeNet foram utilizados os parâmetros: *learning rate* 1e-05, 80.000 *steps*, *lr_policy step*, *gamma* 0.1, *momentum* 0,9, *weight_decay 0.0005*, *transfer learning* [49-50] baseada na ImageNet [31] e configurações do *Pooling* (treino RRM e classificação RMM).

A terceira etapa é responsável por informar o *resultado* da classificação da CNN, ou seja, o resultado do *uso-detecção*. Assim, o usuário informa ao protótipo a imagem a ser testada por PPOBench. A imagem pode estar ofuscada, e como resultado, tem-se a ACC para classificação de imagem pornográfica.

[20] https://caffe.berkeleyvision.org/

6.2.3. Valores de Referência

No sentido de detectar pornografia mediante treinamento de CNN tradicional, inicialmente expõem-se os resultados de dois experimentos. O primeiro é aplicado na base Pu, e o segundo, em Pt. O intuito dos experimentos é o de comparar a detecção de imagem pornográfica no uso de cada base, e assim, analisar se existe impacto no ofuscamento inserido em Pt.

Em ambos os experimentos, as imagens de treinamento e validação de cada base foram utilizadas para criar modelo no uso da CNN CaffeNet [68] e *transfer learning* [49-50]. Para os dois casos, os parâmetros foram os mesmos. O resultado das classificações é apresentado na Tabela 9.

Tabela 9. Resultados: Pu e Pt.

Base	TPR	TNR	FPR	FNR	ACC (%)
Pu	0,9782	0,6741	0,3259	0,0218	75,52
Pt	0,9371	0,5468	0,4532	0,0629	58,46

Conforme visto na Tabela 9, Pu obteve 75,52% de ACC na detecção de pornografia baseando-se nas imagens teste, enquanto Pt obteve 58,46%. Percebe-se que o ofuscamento, que proporciona a ausência das partes íntimas nas imagens de treinamento e validação das imagens pornográficas de Pt, são importantes para a classificação. Com isso, houve diminuição de 17,06 pontos percentuais na ACC.

Conforme a Tabela 9, existe aumento nos valores de FPR e Taxa de Falso Negativo (FNR) em relação a Pu. Isso denota que o ofuscamento inserido confunde a CNN, e no caso de Pt, a CNN não afirma que as imagens pornográficas são normais. Enfim, não é possível afirmar que a causa da diminuição na ACC é responsabilidade do ofuscamento, ou da ausência das partes íntimas.

6.2.4. Reconstituição da Base Pt

As próximas subseções visam responder à questão de pesquisa: é possível aplicar GAN para reconstituição de uma base ofuscada e mesmo assim detectar pornografia? Para tal, é apresentada a reconstituição da base Pt, e após, a detecção de pornografia.

Inicialmente são realizados três experimentos após ter sido verificado na literatura a ampla utilização de GANs, e como elas podem ser aplicadas para a reconstituição de imagens. Assim, inicialmente fez-se a reconstituição de Pt no uso da GAN pix2pixHD. Na Figura 25 são apresentados exemplos de imagens resultantes desta reconstituição.

Mediante análise dos exemplos, percebe-se que a reconstituição não é perfeita. Havendo um bom treinamento, muitas imagens reconstituídas tendem a se aproximar da imagem original. Além disso, percebe-se que as imagens reconstituídas ficam com alguns *pixels* comprometedores, uma espécie de borrão.

Figura 25. Exemplos de imagens de treinamento da base Pt reconstituída.

Durante a execução da reconstituição percebe-se haver uma mudança na distribuição dos *pixels* na imagem reconstituída, e para evitar que isso possa comprometer o segundo experimento, aplica-se a reconstituição em todas as imagens, obtendo-se assim, exemplos de imagens normais que também são submetidos a etapa de reconstituição (Figura 26).

Figura 26. Exemplos de imagens de teste da base Pt reconstituída.

Num segundo experimento é determinada a melhor técnica para reconstituição, e compara-se a reconstituição da base Pt realizada pelas GANs pix2pix e da pix2pixHD. Para tal, é utilizada, por exemplo, a arquitetura CaffeNet [68] usando *transfer learning* [49-50] e parâmetros: 350 épocas, *learning rate* 1e-05, 80.000 *steps*, *lr_policy step*, *gamma* 0.1, *momentum* 0.9, *weight_decay* 0.0005, cujos resultados podem ser vistos na Tabela 10, que apresentam resultados de reconstituição das imagens.

Tabela 10. Resultados: reconstituição das imagens.

GAN	ACC (%)
pix2pix [7]	66,61
pix2pixHD [53]	88,63

Conforme visto na Tabela 10, a pix2pixHD obteve 88,63% de ACC, um aumento de 22,02 pontos percentuais em relação a pix2pix, sendo que ela proporcionou melhor resultado para a reconstituição.

Tabela 11. Resultados.

GAN	TPR	TNR	FPR	FNR	ACC (%)
pix2pixHD [53]	0,9624	0,8318	0,1682	0,0376	88,63

Comparando os resultados apresentados na Tabela 11, com os resultados apresentados para Pu na Tabela 9, verifica-se que houve um aumento em ACC de aproximadamente 13,11 pontos percentuais. Além disso, a TPR diminuiu de 0,97 para 0,9624, assim como FPR diminuiu de 0,3259 para 0,16. Imagina-se que haveria um grande aumento de FN, o que não ocorre. Isto porque todas as imagens foram reconstituídas, e a tendência era a CNN classificar todas as imagens como pornográficas.

Num terceiro experimento, aplica-se pix2pixHD para executar o treinamento em diferentes épocas. Opta-se pelas épocas: 200, 250, 300 e 350, seguindo as mesmas configurações para cada uma (as configurações já foram apresentadas). Quanto mais aprimorado o treinamento, melhor a convergência do modelo gerado pela CNN.

Tabela 12. Resultados: reconstituição das imagens - pix2pixHD [54].

Número de Épocas	ACC (%)
200	78,30
250	88,63
300	86,87
350	86,47

Conforme visto na Tabela 12 de resultados de reconstituição das imagens - pix2pixHD, o melhor resultado apresentado é para a época 250. Quando a reconstituição foi executada com modelo treinado com 350 épocas, ainda assim, melhor resultado que o proporcionado por pix2pix visto na Tabela 11.

6.2.5. Detecção de Pornografia

Objetivando detectar pornografia mediante treinamento de CNN com a inserção/modificação da operação realizada na camada de *Pooling* é implementado a camada R na arquitetura da CNN, i.e., um *Pooling*: uma inserção/alteração realizada na linguagem C++ do *framework* Caffe. O arquivo responsável pelas implementações de *Pooling* do Caffe é editado, onde foi implementado o cálculo de *Max Pooling* e posteriormente, aplicação da raiz quadrada. Ao final da função tem-se o mapa de características.

Para experimentos, a arquitetura de CNN CaffeNet é treinada em diferentes configurações. A CaffeNet possui três *Poolings* que executam *o Max Pooling*. Assim, primeiro foi experimentado a CaffeNet em sua versão original, ou seja, sem alterar suas camadas de *Pooling* (MMM). Após, aplica-se o *Pooling* R na CaffeNet (RMM) para treinamento, e assim sucessivamente. De tal modo que são realizados quatro treinamentos (MMM, RMM, RRM e RRR).

Nos treinamentos são utilizados os seguintes parâmentos no uso da arquitetura CaffeNet [68] e *transfer learning* [49-50]: 250 épocas, *learning rate* 1e-05, 80.000 *steps*, *lr_policy step*, *gamma* 0.1, *momentum* 0.9, *weight_decay* 0.0005.

Posterior ao treinamento, fez-se a classificação, aplicando-se a mesma técnica. A rede treinada com a sequência MMM, é classificada utilizando as configurações MMM, RMM, RRM e RRR.

Na Tabela 13, é observada a inserção/modificação da operação realizada no *Pooling*. Apresentam-se os resultados das classificações quando foram utilizadas as imagens da base Pu.

Quando a CNN é treinada no uso de Pu e com as configurações padrões, ou seja, treino MMM e classificação MMM, obtém-se ACC de 75,52%. Aplicando a abordagem de modificação no *Pooling* proposto, como por exemplo, treino RRR e classificação RMM (86,38%), tem-se um aumento de 10,86 pontos percentuais na detecção de pornografia. A proposição de se treinar com uma configuração, e classificar com outra é estranha. Assim, analisando a Tabela 13, treino RRM e classificação RRM, obtém-se 83,38%.

Tabela 13. Inserção/modificação da operação realizada no *Pooling*: Pu.

Classificação	TPR	TNR	FPR	FPN	ACC (%)	TPR	TNR	FPR	FPN	ACC (%)
	Treino MMM					Treino RMM				
MMM	0,97820	0,67405	0,32595	0,02180	75,52	0,97093	0,69481	0,30519	0,02907	77,56
RMM	0,96810	0,75644	0,24356	0,03190	83,13	0,95957	0,76019	0,23981	0,04043	83,23
RRM	0,97225	0,73668	0,26332	0,02775	81,53	0,9639	0,74468	0,25532	0,0361	82,04
RRR	0,97974	0,69669	0,30331	0,02026	77,90	0,97418	0,70926	0,29074	0,02582	79,04
	Treino RRM					Treino RRR				
MMM	0,97222	0,70799	0,29201	0,02778	78,88	0,9652	0,73964	0,26036	0,0348	81,63
RMM	0,95715	0,78143	0,21857	0,04285	84,84	0,95425	0,80337	0,19663	0,04575	86,38
RRM	0,96101	0,76165	0,23835	0,03899	83,38	0,95996	0,78443	0,21557	0,04004	85,15
RRR	0,97200	0,71941	0,28059	0,02800	79,96	0,97033	0,74979	0,25021	0,02967	82,63

Realizando comparação com o treino MMM e classificação MMM, existem várias configurações interessantes. Em praticamente todos houve uma leve diminuição de FPR.

Tabela 14. Inserção/modificação da operação realizada no *Pooling*: Pt reconstituída. pix2pixHD – 250 épocas.

Classificação	TPR	TNR	FPR	FPN	ACC (%)	TPR	TNR	FPR	FNR	ACC (%)
	Treino MMM					Treino RMM				
MMM	0,96244	0,83176	0,16824	0,03756	88,63	0,95507	0,77221	0,22779	0,04493	84,06
RMM	0,94472	0,88401	0,11599	0,05528	91,21	0,93947	0,84349	0,15651	0,06053	88,56
RRM	0,94689	0,87581	0,12419	0,05311	90,83	0,94514	0,8282	0,1718	0,05486	87,78
RRR	0,95794	0,84544	0,15456	0,04206	89,38	0,95696	0,79828	0,20172	0,04304	86,09
	Treino RRM					Treino RRR				
MMM	0,94092	0,85946	0,14054	0,05908	89,60	0,95101	0,80847	0,19153	0,04899	86,64
RMM	0,92430	0,90216	0,09784	0,07570	91,29	0,93766	0,87236	0,12764	0,06234	90,24
RRM	0,92825	0,89524	0,10476	0,07175	91,11	0,94218	0,86424	0,13576	0,05782	89,94
RRR	0,94029	0,87401	0,12599	0,05971	90,45	0,95386	0,83344	0,16656	0,04614	88,44

Por fim, resolve-se aplicar R no treinamento e classificação das imagens da base Pt reconstituída. Na Tabela 14, é mostrada a inserção/modificação da operação realizada no *Pooling*: Pt reconstituída; pix2pixHD – 250 épocas, apresentam-se os resultados proporcionados pela base Pt reconstituída. Percebe-se que no uso das configurações RRM e RMM, obtém-se ACC de 91,29%. Entretanto, em todos os casos propiciou-se um aumento de FN.

Tabela 15. Resumo – Acurácias: Pu e Pt reconstituída.

	ACC (%)
Pu e MMM MMM	75,52
Pt reconstituída e MMM MMM	88,63
Pt reconstituída e RRM RMM	91,29

Para resumir os resultados apresentados, expõem-se na Figura 27 um gráfico da curva ROC dos três experimentos conforme listados na Tabela 15 que apresenta as acurácias de Pu e Pt reconstituída.

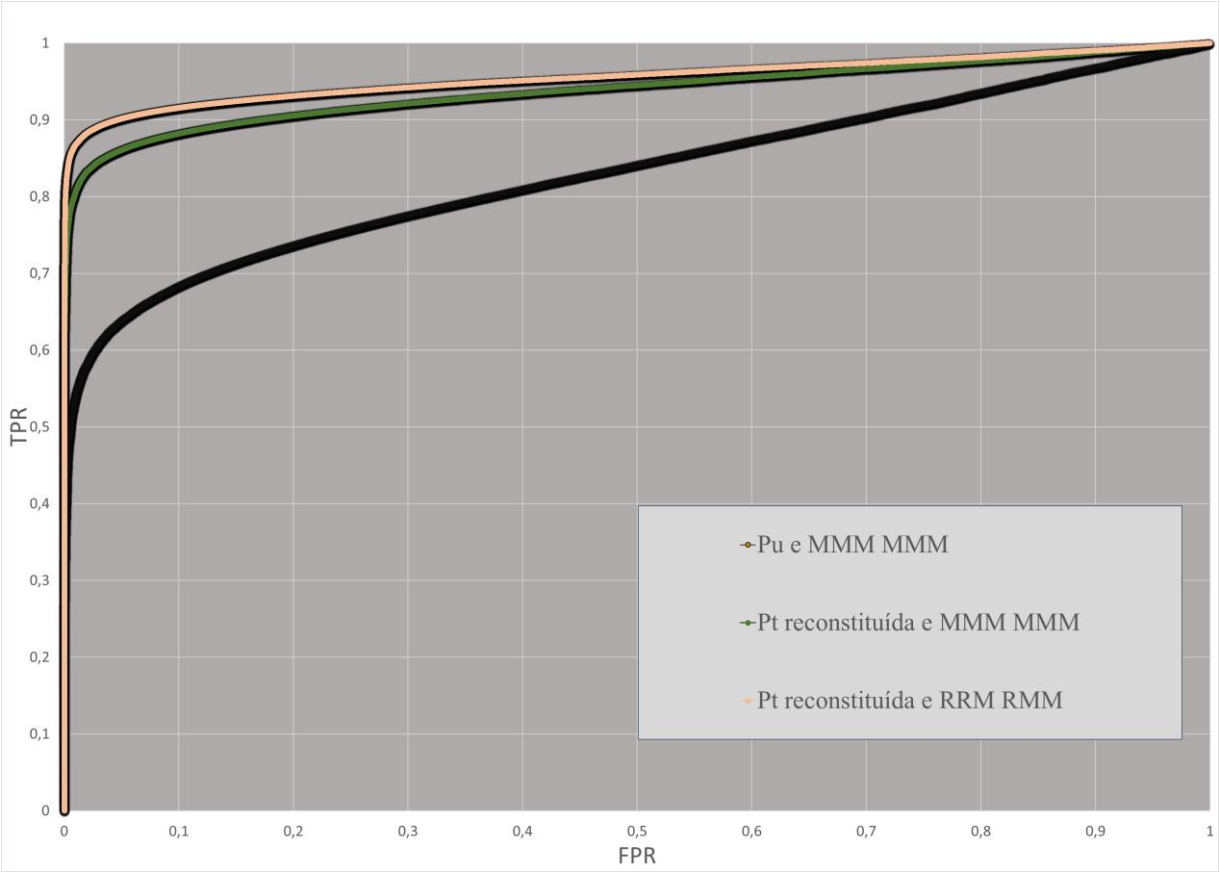


Figura 27. Curva ROC.

Portanto, é constatado que é possível a reconstituição das imagens pornográficas ofuscadas (Pt), destacando-se que a existência de técnicas de CNN que podem proporcionar ainda melhores taxas de acertos [108-109]. Todavia, caso a quantidade e qualidade das imagens utilizadas no treinamento da GAN forem aperfeiçoadas, é provável que os resultados melhorem ainda mais.

O treino da CNN no uso de Pt reconstituída e com a inserção/modificação de cálculo realizado na camada de *Pooling* proporciona uma melhora em termos de ACC. Isso em relação ao treinamento com as imagens originais (Pu) ou mesmo Pt reconstituída no uso do *Max Pooling* original. Percebe-se que, com a reconstituição e a aplicação do cálculo no *Pooling*, o modelo passa a acertar com maior confiança. Assim, é possível o uso de imagens ofuscadas para treinar CNN visando a detecção de imagem pornográfica.

Ademais, para complementar os experimentos, verifica-se a possibilidade de utilização de um *framework* existente na literatura e que faça a visualização dos pontos de ativação que o modelo CNN utiliza para classificar imagens. Para tal, foi escolhido o *framework* LRP *Toolbox for Artificial Neural Networks* (1.3.1)[21].

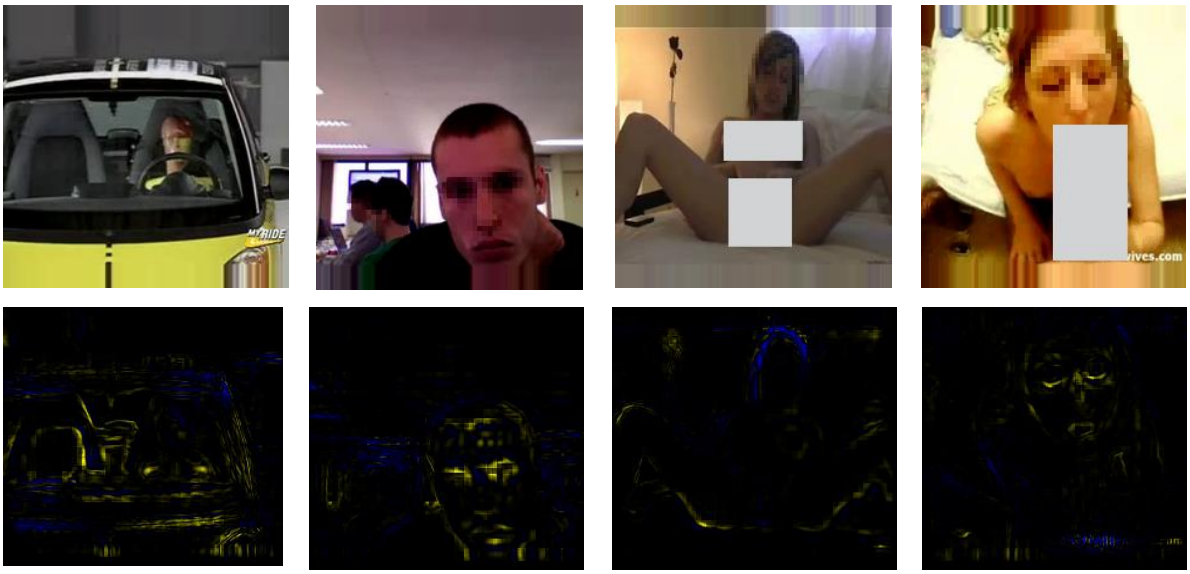


Figura 28. *Explainable* para imagens teste.

[21] https://github.com/sebastian-lapuschkin/lrp_toolbox

Aplica-se os modelos CNNs com as abordagens M e R, tanto para Pu como em Pt reconstituída. Com o fornecimento de imagens de teste não ofuscadas ao *framework* usando o treino (RRM) efetuado com Pt reconstituída e classificação RMM, percebe-se em geral, que as imagens resultantes de ambos os modelos possuem os pontos de atribuição muito parecidos. Em todas as situações, se referem ao contexto. Assim, verifica-se que a CNN pode utilizar partes íntimas para a classificação de imagem pornográfica, mas normalmente, isso é dado pelo contexto, ou seja, uma base de imagens contendo imagens pornográficas ofuscadas realmente esconde as partes íntimas para uma CNN, embora o efeito do ofuscamento atrapalhe.

6.2.6. Discussão

Foram obtidos bons resultados na classificação de imagem pornográfica utilizando modelo treinado por CNN e imagens pornográficas ofuscadas (Pt). PPOBench aumentou a ACC em 15,77% em relação a abordagem com imagens não ofuscadas. Salienta-se que PPOBench contribui com um método que permite o *benchmarking* sem expor publicamente as partes íntimas na base.

Verificou-se que as ferramentas atuais, baseadas em CNN e que fazem a classificação de imagem pornográfica, em geral, não aplicam melhorias na CNN. Ao contrário, em PPOBench é apresentado um método que faz a reconstituição e a inserção/modificação de operação realizada na camada de *Pooling* da CNN. Com essa pequena modificação, PPOBench proporcionou um aumento considerável na taxa de acerto proporcionado pela CNN.

Comprovou-se visualmente o que a literatura afirma: as CNNs realizam a classificação baseando-se no contexto. Verificou-se em testes experimentais, que no caso de imagens pornográficas, isso também ocorre. Imaginava-se a observação apenas das partes íntimas nas imagens geradas pelo *Explainable***,** o que não ocorreu. Talvez se a imagem fornecida ao modelo tivesse um foco maior na parte íntima isso seria um pouco diferente. Enquanto os seres humanos se limitam a classificar imagens pornográficas baseando-se na visualização de objetos, percebe-se que as CNNs se baseiam no contexto.

Capítulo 7

Considerações Finais

Apresentou-se o método proposto em duas partes: *Private Parts Censor* (PPCensor) e PPOBench. O primeiro foi publicado em periódico, e o segundo, está em processo de aperfeiçoamento da escrita do artigo. Ambos se complementam e respondem à questão de pesquisa deste trabalho.

PPCensor é um subproduto que faz a detecção *online* de imagem de pornografia em vídeo. É *user-friendly* e implementado como um servidor *proxy*. Desta forma, transparente e apropriado para dispositivos com recursos limitados. Este foi o primeiro trabalho que trata o conteúdo pornográfico como um problema de detecção de objeto, sendo que se tornou um método semiautomático para ofuscação de partes íntimas detectada em *frames* (imagens).

Contribui-se com uma base publicamente disponível[22] com quatro partes privadas anotadas de um corpo humano (objetos), totalizando 52.215 objetos e denominada de PPO. Ela é a primeira base publicamente disponível na literatura com partes privadas anotadas. Além do mais, pode ser utilizado em diversas aplicações: em escolas, casas, empresas etc., evitando a exposição das partes íntimas de vídeo pornográfico visualizado *online*, diferentemente da literatura que trabalha *offline* e geralmente censura o vídeo todo. A avaliação realizada para um *desktop* com uma configuração típica mostra que o PPCensor apresentou apenas 1,66% de FP no uso da base Pornography-2k e foi capaz de acomodar até 35 conexões *online* de usuários simultâneos.

Destaca-se que a detecção de objetos faz a ligação entre PPCensor e PPOBench, que por sua vez, é um método para publicação de imagens de pornografia sem revelar as partes íntimas nas cenas, ou seja, um método de detecção de pornografia. Ele faz a reconstituição de

[22] https://secplab.ppgia.pucpr.br/?q=ppcensor

imagens ofuscadas e a inserção/modificação de operação realizada na camada de *Pooling* da CNN. O método é importante e útil para a solução, inclusive de problemas internacionais. Permite-se o *benchmarking* sem expor publicamente as partes íntimas na base. Para comprovação de PPOBench, foi formalizada a base de imagens (Pt), a qual foi dividida em duas classes e é composta por 409.130 imagens, sendo que as imagens de treinamento e validação da classe pornográfica estão ofuscadas.

Em resultados experimentais, PPOBench proporcionou um aumento considerável na taxa de acerto. Reconstituindo imagens ofuscadas com a pix2pixHD e treinamento com CNN tendo a camada de *Pooling* modificada obteve-se 15,77 pontos percentuais a mais do que a CNN tradicional treinada com Pu. Ainda, na avaliação foi verificado que as CNNs não se baseiam apenas na visualização das PPO ao realizarem a detecção de pornografia. As CNNs se utilizam do contexto para classificar as imagens pornográficas. Tal afirmação é dada a partir dos resultados apresentados na subseção 6.2.5.

As dificuldades para realização deste trabalho foram várias, sendo uma das principais, o tempo demandado pela falta de base de imagens pornográficas e pública. Por exemplo: foi necessário pré-processar os 2.000 vídeos da Pornography-2k. Eles tiveram suas imagens extraídas em um intervalo de 10 imagens, totalizando 1.376.035 imagens, nos quais 389.808 são normais e 986.233 normais. Todas as imagens foram submetidas a pré-processamento.

Como trabalhos futuros indica-se melhorias na base PPO, possibilitando o alcance de maiores taxas de mAP. Outrora, o uso de outras arquiteturas de CNN em ambos os métodos apresentados neste trabalho. Assim como, melhor pré-processamento nas imagens que foram utilizadas em PPOBech: existem imagens que possuem um foco nas partes íntimas, tornando a proporção do ofuscamento em relação a imagem muito grande. Outrora, poder-se-ia melhorar a base PPO adicionando outras classes por exemplo. Assim como realizar experimentos a partir de outras imagens ofuscadas.

Na prática, os *Internet Security Suites* (ISS) podem utilizar controle paterno para bloquear conteúdos ilegais/impróprios, usando para tal uma listagem de URLs identificadas com este tipo de conteúdo. No melhor caso, um conteúdo pornográfico fica acessível num curto lapso de tempo entre a sua divulgação numa URL e a sua identificação pelos ISS. PPCensor poderia ser modelado e utilizado na detecção de Pornografia Infantil (PI), fornecendo as URLs

suspeitas para instituições como a NCMEC[23] (*National Center for Missing Exploited Children*). Também como trabalhos futuros, considera-se a aplicação de outros detectores de objeto na base PPO, como por exemplo, YOLO [48].

Por fim, para suprimir problemas para a Pontifícia Universidade Católica do Paraná (PUC-PR), para execução deste trabalho foi registrado no Comitê de Ética e Pesquisa (CEP) da PUC-PR (Anexo I).

[23] https://www.missingkids.org/HOME

Referências Bibliográficas

[1] H. Enough, “Pornography Statistics.” https://enough.org/stats_porn_industry (accessed Feb. 11, 2021).

[2] Statista, “How Much of the Internet Consists of Porn?” https://www.statista.com/chart/16959/share-of-the-internet-that-is-porn/ (accessed Feb. 11, 2021).

[3] Pornhub, “2018 Year in Review.” https://www.pornhub.com/insights/ (accessed Feb. 11, 2021).

[4] S. Karamizadeh and A. Arabsorkhi, “Methods of Pornography Detection: Review,” in *Proceedings of the 10th International Conference on Computer Modeling and Simulation*, 2018, pp. 33–38, doi: 10.1145/3177457.3177484.

[5] W. Rawat and Z. Wang, “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review,” *Neural Comput.*, vol. 29, no. 9, pp. 2352–2449, 2017, doi: 10.1162/neco_a_00990.

[6] A. R. Pathak, M. Pandey, and S. Rautaray, “Application of Deep Learning for Object Detection,” *Procedia Comput. Sci.*, vol. 132, pp. 1706–1717, 2018, doi: 10.1016/j.procs.2018.05.144.

[7] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, “Image-to-Image Translation with Conditional Adversarial Networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5967–5976, doi: 10.1109/CVPR.2017.632.

[8] M. Moustafa, “Applying deep learning to classify pornographic images and videos,” *7th Pacific-Rim Symposium on Image and Video Technology (PSIVT)*, 2015.

[9] J. Mahadeokar and G. Pesavento, “Open sourcing a deep learning solution for detecting nsfw imagese,” 2016. https://yahooeng.tumblr.com/post/151148689421/open-sourcing-a-deep-learning-solution-for (accessed Feb. 11, 2021).

[10] M. Perez, S. Avila, D. Moreira, D. Moraes, V. Testoni, E. Valle, S. Goldenstein and A. Rocha, “Video pornography detection through deep learning techniques and motion information,” *Neurocomputing*, vol. 230, pp. 279–293, 2017, doi: https://doi.org/10.1016/j.neucom.2016.12.017.

[11] J. Wehrmann, G. S. Simões, R. C. Barros, and V. F. Cavalcante, “Adult content detection in videos with convolutional and recurrent neural networks,” *Neurocomputing*, vol. 272, pp. 432–438, 2018, doi: https://doi.org/10.1016/j.neucom.2017.07.012.

[12] P. Vitorino, S. Avila, M. Perez, and A. Rocha, “Leveraging deep neural networks to fight child pornography in the age of social media,” *Journal of Visual Communication and*

Image Representation (JVCI'18), vol. 50, pp. 303–313, 2018, doi: 10.1016/j.jvcir.2017.12.005.

[13] J. Macedo, F. Costa, and J. A. dos Santos, "A Benchmark Methodology for Child Pornography Detection," in *31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, 2018, pp. 455–462, doi: 10.1109/SIBGRAPI.2018.00065.

[14] F. Idris and S. Panchanathan, "Review of image and video indexing techniques," *Journal of Visual Communication and Image Representation*, 1997, 8(2), pp. 146-166.

[15] Y. Kuroki, T. Nish, S. Kobayashi, H. Oyaizu and S. Yoshimura, "A psychophysical study of improvements in motion-image quality by using high frame rates," *Journal of the Society for Information Display*, 15, 1 (2007), 61, doi: 10.1889/1.2451560.

[16] Wearesocial, "Digital 2021: the latest insights into the 'state of digital'." https://wearesocial.com/blog/2021/01/digital-2021-the-latest-insights-into-the-state-of-digital (accessed April. 11, 2021).

[17] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M. Shyu, S. Chen, and S. S. Iyengar, "A Survey on Deep Learning: Algorithms, Techniques, and Applications," *ACM Comput. Surv.*, vol. 51, no. 5, 2018, doi: 10.1145/3234150.

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

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

[20] I. H. Witten, E. Frank, M. A. Hall, C. J. PAL, *Practical Machine Learning Tools and Techniques*. Editora Morgan Kaufmann. 2016. 4th Edition. ISBN: 9780128042915.

[21] J. Mallmann, A. O. Santin, E. K. Viegas, R. R. dos Santos, and J. Geremias, "PPCensor: Architecture for real-time pornography detection in video streaming," *Future Generation Computer System*, vol. 112, pp. 945–955, 2020, doi: 10.1016/j.future.2020.06.017.

[22] J. Mallmann, A. O. Santin, A. Britto Jr and R. R. dos Santos, "Mitigando os Efeitos de GAN em Classificação de Imagens," In*: XIX SBSeg - Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg)*, São Paulo. 2019.

[23] J. Mallmann, A.O. Santin, E. K. Viegas, R. R. dos Santos, and J. Geremias, "Ferramenta PPCensor: detecção de pornografia em tempo real no streaming de vídeo," In*: XX Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)* - Salão de Ferramentas, 2020. pp. 1-5.

[24] R. R. Santos, E. K. Viegas, A.O. Santin, J. Mallmann, "Sistema de Detecção de Intrusão Baseado em Aprendizagem por Reforço," In*: XX Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)*, 2020.

[25] F. Chollet, *Deep Learning with Python*. Manning Publications, 2018.

[26] Yuming Hua, Junhai Guo, and Hua Zhao, "Deep Belief Networks and deep learning," in *Proceedings of 2015 International Conference on Intelligent Computing and Internet of Things*, Jan. 2015, pp. 1–4, doi: 10.1109/ICAIOT.2015.7111524.

[27] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," *CoRR*, vol. abs/1406.1, 2014, [Online]. Available: http://arxiv.org/abs/1406.1078.

[28] C. Goller and A. Küchler, "Learning task-dependent distributed representations by backpropagation through structure," in *Proceedings of International Conference on Neural Networks (ICNN'96)*, Washington, DC, USA, June 3-6, 1996, pp. 347–352, doi: 10.1109/ICNN.1996.548916.

[29] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 677–691, 2017, doi: 10.1109/TPAMI.2016.2599174.

[30] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 2014, pp. 2672–2680.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, 2012, pp. 1097–1105.

[32] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[33] A. Karpathy, "CS231n: Convolutional Neural Networks for Visual Recognition," 2016. http://cs231n.github.io/convolutional-networks/ (accessed Mar. 20, 2021).

[34] M. A. Ponti, L. S. F. Ribeiro, T. S. Nazare, T. Bui, and J. Collomosse, "Everything You Wanted to Know about Deep Learning for Computer Vision but Were Afraid to Ask," in *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, pp. 17–41, doi: 10.1109/SIBGRAPI-T.2017.12.

[35] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, 2010, pp. 807–814.

[36] Y.-L. Boureau, J. Ponce, and Y. Lecun, "A theoretical analysis of feature pooling in vision algorithms," 2010.

[37] N. Akhtar and U. Ragavendran, "Interpretation of intelligence in CNN-pooling

processes: a methodological survey," *Neural Computing Application*, vol. 32, no. 3, pp. 879–898, 2020, doi: 10.1007/s00521-019-04296-5.

[38] Y. LeCun, "Generalization and Network Design Strategies," in *Connectionism in Perspective*, 1989.

[39] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, "Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3," in *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*, 2019, pp. 1–6, doi: 10.1109/UVS.2019.8658300.

[40] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," *APSIPA Trans. Signal Inf. Process.*, vol. 3, p. e2, 2014, doi: 10.1017/atsip.2013.9.

[41] S. Srinivas, R. K. Sarvadevabhatla, K. R. Mopuri, N. Prabhu, S. S. S. Kruthiventi, and R. V. Babu, "A Taxonomy of Deep Convolutional Neural Nets for Computer Vision," *Front. Robot. AI*, vol. 2, p. 36, 2016, doi: 10.3389/frobt.2015.00036.

[42] S. Lapuschkin, A. Binder, G. Montavon, K.-R. Müller, and W. Samek, "The LRP Toolbox for Artificial Neural Networks," *J. Mach. Learn. Res.*, vol. 17, no. 114, pp. 1–5, 2016, [Online]. Available: http://jmlr.org/papers/v17/15-618.html.

[43] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?': Explaining the Predictions of Any Classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144, doi: 10.1145/2939672.2939778.

[44] Z. Zhao, P. Zheng, S. Xu, and X. Wu, "Object Detection With Deep Learning: A Review," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.

[45] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3296–3297, doi: 10.1109/CVPR.2017.351.

[46] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017, doi: 10.1109/TPAMI.2016.2577031.

[47] W. Liu, W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," in *Computer Vision -- ECCV*, 2016, *Springer International Publishing*, pp. 21–37.

[48] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement.", CoRR, 2018.

[49] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1717–1724, doi: 10.1109/CVPR.2014.222.

[50] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A Survey on Deep Transfer Learning,” *CoRR*, vol. abs/1808.0, 2018, [Online]. Available: http://arxiv.org/abs/1808.01974.

[51] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and L. Zitnick, “Microsoft COCO: Common Objects in Context,” in *Computer Vision -- ECCV 2014*, 2014, pp. 740–755.

[52] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, “Generative Adversarial Networks: An Overview,” *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 53–65, Jan. 2018, doi: 10.1109/MSP.2017.2765202.

[53] Y. Hong, U. Hwang, J. Yoo, and S. Yoon, “How Generative Adversarial Networks and Their Variants Work: An Overview,” *ACM Comput. Surv.*, vol. 52, no. 1, 2019, doi: 10.1145/3301282.

[54] T. Wang, M. Liu, J. Zhu, A. Tao, J. Kautz, and B. Catanzaro, “High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8798–8807, doi: 10.1109/CVPR.2018.00917.

[55] K. He, X. Zhang, S. Ren, and J. Sun. 2015. “Deep residual learning for image recognition,” in *Proc. CoRR*, 2015.

[56] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[57] K. Simonvan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition”. *CoRR*, vol. abs/1409.1556, 2014.

[58] C. A. Poyton, *Digital Video and HD Algorithms and Interfaces*, 2ª. Edição. Morgan Kaufmann, 2012.

[59] D. L. Brinkley and R. R. Schell, “Concepts and terminology for computer security,”. In: ABRAMS, M. D.; JAJODIA, S.; PODELL, H. J. (Ed.). Information security: an integrated collection of essays. Los Alamitos, CA: *IEEE Computer Society Press*, jan. 1995. p. 40-97.

[60] J. B. D. Joshi, W. G. Aref, A. Ghafoor, and E. H. Spafford, “Security Models for Web-Based Applications,” *Commun. ACM*, vol. 44, no. 2, pp. 38–44, 2001, doi: 10.1145/359205.359224.

[61] J. F. Kurose and K. Ross, *Redes de Computadores e a Internet: Uma Abordagem Top-Down*, 6ª. Edição. Editora Pearson Education do Brasil, 2013.

[62] A. S. Tanenbaum and D. Wetherall, *Redes de Computadores*. 5ª. Edição, São Paulo: Pearson, 2011.

[63] S. E. Umbaugh, *Digital image processing and analysis: human and computer vision*

applications with CVIPtools, 2ª. Edição. 1998.

[64] R.C. Gonzalez and R. E. Woods, *Digital image processing. Editora Prentice Hall.*, 2ª. Edição. 1992.

[65] M. Polastro and 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.

[66] D. Moreira, S. Avila, M. Perez, D. Moraes, V. Testoni, E. Valle, S. Goldenstein, and A. Rocha, "Pornography classification: The hidden clues in video space–time," *In Forensic Science International*, vol. 268, pp. 46–61, 2016, doi: https://doi.org/10.1016/j.forsciint.2016.09.010.

[67] K. Soomro, A. R. Zamir, and S. Mubarak, "*UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild*," *CRCV-TR-12-01*, November, 2012.

[68] Y. Jia, Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding," in *Proceedings of the 22nd ACM International Conference on Multimedia*, 2014, pp. 675–678, doi: 10.1145/2647868.2654889.

[69] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017, doi: 10.1145/3065386.

[70] T. Kim and J. Yang, "Selective Feature Anonymization for Privacy-Preserving Image Data Publishing," *Electronics*, vol. 9, no. 5, 2020, doi: 10.3390/electronics9050874.

[71] T. Kim and J. Yang, "Latent-Space-Level Image Anonymization With Adversarial Protector Networks," *IEEE Access*, vol. 7, pp. 84992–84999, 2019, doi: 10.1109/ACCESS.2019.2924479.

[72] A. Odena, C. Olah, and J. Shlens, "Conditional Image Synthesis with Auxiliary Classifier GANs," *arXiv* 2016, arXiv:1610.09585.

[73] M. D. More, D. M. Souza, J. Wehrmann, and R. C. Barros, "Seamless Nudity Censorship: an Image-to-Image Translation Approach based on Adversarial Training," in *International Joint Conference on Neural Networks (IJCNN)*, 2018, pp. 1–8, doi: 10.1109/IJCNN.2018.8489407.

[74] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2242–2251, doi: 10.1109/ICCV.2017.244.

[75] G. S. Simões, J. Wehrmann, and R. C. Barros, "Attention-based Adversarial Training for Seamless Nudity Censorship," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019, pp. 1–8, doi: 10.1109/IJCNN.2019.8851849.

[76] I. Joshi, A. Anand, M. Vatsa, R. Singh, S. D. Roy, and P. Kalra, "Latent Fingerprint

Enhancement Using Generative Adversarial Networks," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2019, pp. 895–903, doi: 10.1109/WACV.2019.00100.

[77] H. Hukkelås, R. Mester, and F. Lindseth, "DeepPrivacy: A Generative Adversarial Network for Face Anonymization," In: Bebis G. et al. (eds) Advances in *Visual Computing*. *ISVC 2019*. Lecture Notes in Computer Science, vol 11844. Springer, Cham. DOI: 10.1007/978-3-030-33720-9_44.

[78] J. Li, Y. Wang, C. Wang, Y. Tai, J. Qian, J. Yang, C. Wang, J. Li, and F. Huang, "DSFD: Dual Shot Face Detector," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5055–5064, doi: 10.1109/CVPR.2019.00520.

[79] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," in *Advances in Neural Information Processing Systems*, 2017, vol. 30, pp. 6626–6637, [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf.

[80] E. Piacentino and C. Angulo, "Anonymizing Personal Images Using Generative Adversarial Networks," in *Bioinformatics and Biomedical Engineering*, 2020, pp. 395–405. doi: 10.1007/978-3-030-45385-5_35.

[81] E. Piacentino and C. Angulo, "Generating Fake Data Using GANs for Anonymizing Healthcare Data," in *Bioinformatics and Biomedical Engineering*, 2020, pp. 406–417. doi: 10.1007/978-3-030-45385-5_36.

[82] F. Nian, T. Li, Y. Wang, M. Xu, and J. Wu, "Pornographic Image Detection Utilizing Deep Convolutional Neural Networks," *Neurocomput.*, vol. 210, no. C, pp. 283–293, 2016, doi: 10.1016/j.neucom.2015.09.135.

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

[84] X. Ou, H. Ling, H. Yu, P. Li, F. Zou, and S. Liu, "Adult Image and Video Recognition by a Deep Multicontext Network and Fine-to-Coarse Strategy," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 5, 2017, doi: 10.1145/3057733.

[85] S. Avila, N. Thome, M. Cord, E. Valle, and A. de A. Araújo, "Pooling in image representation: The visual codeword point of view," *Comput. Vis. Image Underst.*, vol. 117, no. 5, pp. 453–465, 2013, doi: https://doi.org/10.1016/j.cviu.2012.09.007.

[86] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9, doi: 10.1109/CVPR.2015.7298594.

[87] Z. Fu, J. Li, G. Chen, T. Yu, and T. Deng, "PornNet: A Unified Deep Architecture for Pornographic Video Recognition" in 2021 *Appl. Sci.* 11, no. 7: 3066. https://doi.org/10.3390/app11073066.

[88] A. Gangwar, V. González-Castro, E. Alegre, and E. Fidalgo, "AttM-CNN: Attention and metric learning based CNN for pornography, age and Child Sexual Abuse (CSA) Detection in images," Neurocomputing, Volume 445, 2021, Pages 81-104, https://doi.org/10.1016/j.neucom.2021.02.056.

[89] L. Wang, J. Liao, and C. Xu, "Vehicle Detection Based on Drone Images with the Improved Faster R-CNN," in *Proceedings of the 11th International Conference on Machine Learning and Computing*, 2019, pp. 466–471, doi: 10.1145/3318299.3318383.

[90] G. Plastiras, C. Kyrkou, and T. Theocharides, "Efficient ConvNet-Based Object Detection for Unmanned Aerial Vehicles by Selective Tile Processing," 2018, doi: 10.1145/3243394.3243692.

[91] P. Maheshwari, D. Alex, S. Banerjee, S. Behera, and S. Panda, "Top View Person Detection and Counting for Low Compute Embedded Platforms," in *Proceedings of the 2nd International Conference on Video and Image Processing*, 2018, pp. 35–43, doi: 10.1145/3301506.3301548.

[92] S. Saikia, E. Fidalgo, E. Alegre, and L. Fernández-Robles, "Object Detection for Crime Scene Evidence Analysis Using Deep Learning". In: Battiato S., Gallo G., Schettini R., Stanco F. (eds) Image Analysis and Processing - *ICIAP 2017*. Lecture Notes in Computer Science, vol 10485. Springer, Cham.

[93] R. Zhao, W. Ouyang, H. Li, and X. Wang, "Saliency detection by multi-context deep learning," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1265–1274, doi: 10.1109/CVPR.2015.7298731.

[94] C. Lee, P. Gallagher, and Z. Tu, "Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree". *CoRR* 2015, abs/1509.08985.

[95] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," in *Computer Vision -- ECCV*, 2014, pp. 346–361.

[96] Z. Zhong, L. Jin, and Z. Feng, "Multi-Font Printed Chinese Character Recognition Using Multi-Pooling Convolutional Neural Network," in *Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR)*, 2015, pp. 96–100, doi: 10.1109/ICDAR.2015.7333733.

[97] M. Zeiler and R. Fergus, "Stochastic Pooling for Regularization of Deep Convolutional Neural Networks," *CoRR*, 2013, abs/1301.3557.

[98] B. Graham, "Fractional Max-Pooling," *CoRR*, 2014, abs/1412.6071.

[99] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up Convolutional Neural Networks with Low Rank Expansions," *CoRR*, 2014, abs/1405.3866.

[100] R. Girshick, "Fast R-CNN," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.

[101] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *CoRR*, vol. abs/1311.2, 2013, [Online]. Available: http://arxiv.org/abs/1311.2524.

[102] M. Sun, Z. Song, X. Jiang, J. Pan, and Y. Pang, “Learning Pooling for Convolutional Neural Network,” *Neurocomputing*, vol. 224, no. C, pp. 96–104, 2017, doi: 10.1016/j.neucom.2016.10.049.

[103] Z. Chen, J. Lin, V. Chandrasekhar, and L. Duan, “Gated Square-Root Pooling for Image Instance Retrieval,” in *25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 1982–1986, doi: 10.1109/ICIP.2018.8451486.

[104] A. M. Romano and A. A. Hernandez, “Improved Pooling Scheme for Convolutional Neural Networks,” *7th International Conference on Information, Communication and Networks (ICICN)*, Macao, 2019, pp. 201-206, doi: 10.1109/ICICN.2019.8834960.

[105] Squid, “Squid Development Projects.” http://devel.squid-cache.org/. (accessed Feb. 11, 2021).

[106] “Library to download YouTube content and retrieve metadata.” https://pythonhosted.org/pafy/ (accessed Feb. 11, 2021).

[107] TensorFlow, “Tensorflow Object Detection API.” https://www.tensorflow.org/ (accessed Feb. 12, 2021).

[108] D. Sil, A. Dutta, and A. Chandra, “Convolutional Neural Networks for Noise Classification and Denoising of Images,” in *TENCON - IEEE Region 10 Conference (TENCON)*, 2019, pp. 447–451, doi: 10.1109/TENCON.2019.8929277.

[109] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila, “Noise2Noise: Learning image restoration without clean data,” in *Proceedings of the 35th International Conference on Machine Learning*, 2018, vol. 80, pp. 2965–2974, http://proceedings.mlr.press/v80/lehtinen18a.html. (accessed Mar. 30, 2021).

Anexo I

Comitê de Ética em Pesquisa da PUCPR

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ - PUC/ PR

Plataforma Brasil

COMPROVANTE DE ENVIO DO PROJETO

DADOS DO PROJETO DE PESQUISA

Título da Pesquisa: DETECÇÃO INTELIGENTE EM TEMPO REAL DE URLs QUE DISSEMINAM IMAGEM DE PORNOGRAFIA INFANTIL EM VÍDEOS

Pesquisador: JACKSON MALLMANN

Versão: 1

CAAE: 12087119.1.0000.0020

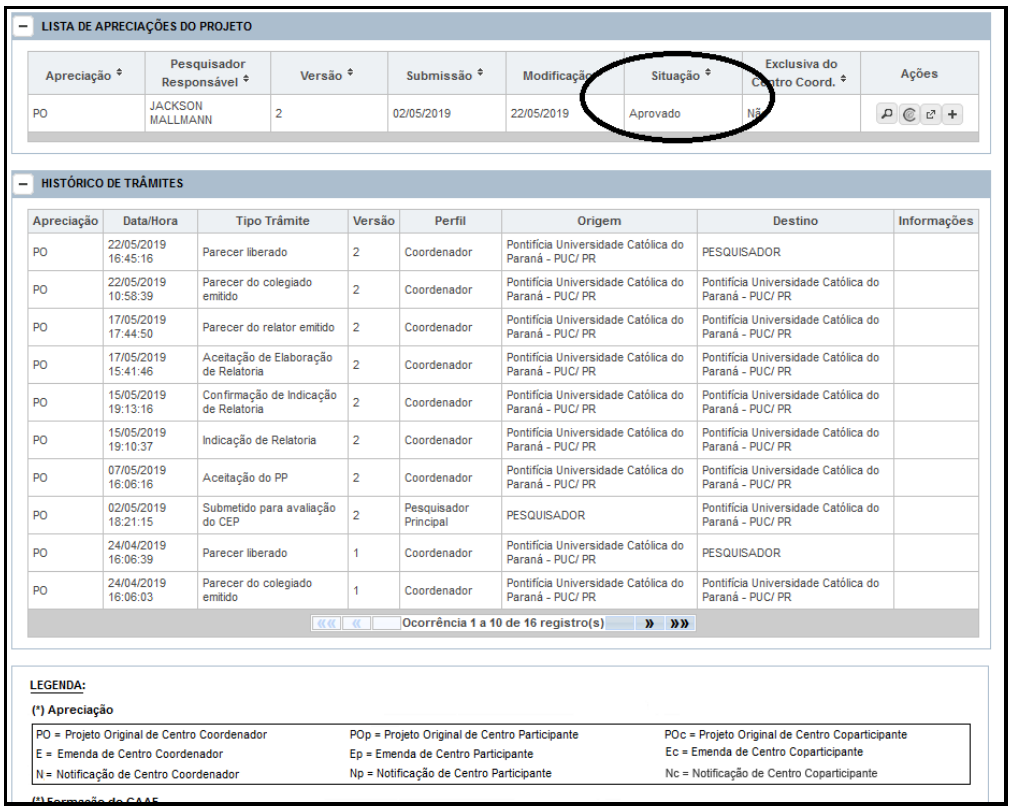
Instituição Proponente: Pontifícia Universidade Católica do Parana - PUCPR

DADOS DO COMPROVANTE

Número do Comprovante: 041480/2019

Patrocionador Principal: Financiamento Próprio

Informamos que o projeto DETECÇÃO INTELIGENTE EM TEMPO REAL DE URLs QUE DISSEMINAM IMAGEM DE PORNOGRAFIA INFANTIL EM VÍDEOS que tem como pesquisador responsável JACKSON MALLMANN, foi recebido para análise ética no CEP Pontifícia Universidade Católica do Paraná - PUC/ PR em 17/04/2019 às 15:59.



LISTA DE APRECIAÇÕES DO PROJETO

Apreciação	Pesquisador Responsável	Versão	Submissão	Modificação	Situação	Exclusiva do Centro Coord.	Ações
PO	JACKSON MALLMANN	2	02/05/2019	22/05/2019	Aprovado	Não	

HISTÓRICO DE TRÂMITES

Apreciação	Data/Hora	Tipo Trâmite	Versão	Perfil	Origem	Destino	Informações
PO	22/05/2019 16:45:16	Parecer liberado	2	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	PESQUISADOR	
PO	22/05/2019 10:58:39	Parecer do colegiado emitido	2	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	Pontifícia Universidade Católica do Paraná - PUC/ PR	
PO	17/05/2019 17:44:50	Parecer do relator emitido	2	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	Pontifícia Universidade Católica do Paraná - PUC/ PR	
PO	17/05/2019 15:41:46	Aceitação de Elaboração de Relatoria	2	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	Pontifícia Universidade Católica do Paraná - PUC/ PR	
PO	15/05/2019 19:13:16	Confirmação de Indicação de Relatoria	2	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	Pontifícia Universidade Católica do Paraná - PUC/ PR	
PO	15/05/2019 19:10:37	Indicação de Relatoria	2	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	Pontifícia Universidade Católica do Paraná - PUC/ PR	
PO	07/05/2019 16:06:16	Aceitação do PP	2	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	Pontifícia Universidade Católica do Paraná - PUC/ PR	
PO	02/05/2019 18:21:15	Submetido para avaliação do CEP	2	Pesquisador Principal	PESQUISADOR	Pontifícia Universidade Católica do Paraná - PUC/ PR	
PO	24/04/2019 16:06:39	Parecer liberado	1	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	PESQUISADOR	
PO	24/04/2019 16:06:03	Parecer do colegiado emitido	1	Coordenador	Pontifícia Universidade Católica do Paraná - PUC/ PR	Pontifícia Universidade Católica do Paraná - PUC/ PR	

Ocorrência 1 a 10 de 16 registro(s)

LEGENDA:

(*) Apreciação

PO = Projeto Original de Centro Coordenador	POp = Projeto Original de Centro Participante	POc = Projeto Original de Centro Coparticipante
E = Emenda de Centro Coordenador	Ep = Emenda de Centro Participante	Ec = Emenda de Centro Coparticipante
N = Notificação de Centro Coordenador	Np = Notificação de Centro Participante	Nc = Notificação de Centro Coparticipante

(*)Formação do CAAE