Pipeline de Dados para Detecção de Anomalias em Vídeos de Cena Única ⋆

Fábio Ricardo Oliveira Bento * **Raquel Frizera Vassallo** **
Jorge Leonid Aching Samatelo **

* *Coordenadoria de Eletrotécnica, Instituto Federal do Espírito Santo, Guarapari, ES (e-mail: fbento@ifes.edu.br)*
** *Departamento de Engenharia Elétrica, Universidade Federal do Espírito Santo, Vitória, ES (e-mail: raquel@ele.ufes.br, jorge.samatelo@ufes.br)*

Abstract: Data pipeline consists of a sequence of actions that preprocess and extract information from datasets. In the context of anomaly detection, the data pipeline has application in the availability of structured and relevant information for the detection task. This article proposes an approach to the problem of detecting anomalies in single-scene video based on a data pipeline, composed of two parts: a *patches* extractor and a patches classification model. We performed experiments on the *Street Scene* dataset, achieving AUC = 0.898 and AUPRC = 0.916, which are results compatible with the current literature.

Resumo: *Pipeline* de dados consiste em uma sequência de ações que preprocessam e extraem informações de conjuntos de dados. No contexto de detecção de anomalias, *pipeline* de dados tem aplicação na disponibilidade de informações estruturadas e relevantes para a tarefa de detecção. Nesse artigo é proposta uma abordagem para o problema de detecção de anomalias em vídeo de cena única baseado em um *pipeline* de dados com localização espacial dos eventos anômalos, composto de duas partes: um extrator de *patches* e um modelo de classificação de *patches*. Foram realizados experimentos no conjunto de dados *Street Scene*, os quais foram avaliados pelas métricas AUC e AUPRC que são, respectivamente, a área abaixo da curva ROC (*Receiver Operating Characteristic Curve*), e a área abaixo da curva *Precision* vs. *Recall*. Foram obtidos AUC = 0.898 e AUPRC = 0.916, os quais são resultados compatíveis com a literatura atual.

Keywords: Smart Cities, Computer Vision, Deep Learning, Anomaly Detection.
Palavras-chaves: Cidades Inteligentes, Visão Computacional, Aprendizagem Profunda, Detecção de Anomalias.

1. INTRODUÇÃO

O monitoramento automático de eventos anômalos, através de vídeo, pode contribuir para a melhoria da segurança em vias urbanas (Santhosh et al., 2020). No entanto, é importante contextualizar a abordagem de detecção de anomalias em função da quantidade de cenas envolvidas, da disponibilidade de dados durante treinamento e do nível de protocolo de avaliação.

A quantidade de cenas envolvidas na detecção de anomalias em vídeo pode ser categorizada em cena única (com câmera em posição fixa) ou em cenas múltiplas (Ramachandra et al., 2020). Essa distinção é importante pois a análise de anomalias requer que tais eventos sejam contextualizados de forma espaço-temporal, ou seja, depende da cena.

A disponibilidade de dados rotulados e eventos anormais durante o período de treinamento do modelo também é relevante pois, na prática, ocorrem apenas escassos eventos anômalos passíveis de rotulação. No entanto, mesmo uma fração dos raros eventos anômalos observados podem ajudar a detectar anomalias idênticas ou similares no futuro. Portanto, seguindo Scheirer et al. (2012); Liu et al. (2019), define-se uma abordagem de detecção supervisionada de anomalias em conjunto aberto (*open-set supervised*) onde apenas uma fração dos escassos eventos anômalos observados, e numerosos eventos normais estão disponíveis durante o treinamento.

O nível de protocolo de avaliação, por sua vez, define se os eventos serão monitorados em nível de *frame* ou em nível de *patch* [1]. O nível selecionado determina a possibilidade de analisar a localização espacial de anomalias e, portanto, classificar corretamente múltiplos eventos anômalos em um *frame*.

Nesse contexto, o presente trabalho tem foco em detecção supervisionada de anomalias em conjunto aberto, com

⋆ Trabalho realizado com o apoio do IFES - Instituto Federal do Espírito Santo através do projeto "Videomonitoramento Inteligente Aplicado à Segurança Pública" cadastrado sob o número: 23183.001228/2020-70, da FAPES - Fundação de Amparo a Pesquisa e Inovação do Espırito Santo através do Projeto 577/2018, e da NVIDIA Corporation através da doação da GPU Titan V.

[1] *Patch* pode ser definido como um subconjunto dos pixels pertencente a determinado *frame*.

protocolo em nível de *patch*, em vídeos de cena única. Durante esse trabalho identificou-se que um *pipeline* de dados bem estruturado é essencial para o sucesso do modelo de detecção de anomalias em vídeo. Portanto, foram desenvolvidas várias etapas sequenciais que extraem, preprocessam e transformam os dados, fornecendo *patches* para o treinamento e teste de um modelo de classificação.

Trabalhos recentes utilizam cenas múltiplas e protocolo em nível de *frame* (Sultani et al., 2018; Dubey et al., 2019; Zhong et al., 2019). No entanto, detectar anomalias em vídeos de cenas múltiplas pode requerer conhecimento prévio e atualização periódica de parâmetros das câmeras. O protocolo em nível de *frame*, por sua vez, não observa a localização espacial dos eventos anômalos e pode conduzir a análises equivocadas. Atualmente já é muito comum na literatura a utilização de modelos de aprendizado de máquina do tipo redes neurais profundas, que comumente apresentam os resultados mais promissores em temáticas de visão computacional. Este trabalho aborda o problema em estudo com redes neurais profundas (Liu et al., 2019; Yoshihashi et al., 2019). Tais modelos são iterativos, pois cada etapa é repetida continuamente para melhorar a precisão do resultado. Portanto, um *pipeline* de dados, que codifique e automatize o fluxo de dados a serem processados pelo modelo, é essencial para o sucesso da detecção de anomalias em vídeo.

Portanto, nesse trabalho adotou-se uma abordagem em cena única, que não requer calibração de câmera ou informações adicionais. Utilizou-se um protocolo de avaliação em nível de *patch*, que considera a localização espacial das anomalias, e permite que múltiplos eventos anômalos sejam corretamente classificados em um *frame*. Além disso, desenvolveu-se um *pipeline* de dados que baseia-se no seccionamento dos vídeos em clipes [2], fusão da informação espaço-temporal, detecção de pontos de interesse e extração *patches*. Os *patches* foram utilizados para treinar um modelo de classificação neural orientado à detecção de anomalias. Tal modelo é composto por um extrator de características constituído por uma rede neural convolucional pré-treinada seguida de camadas totalmente conectadas, que, em última instância, efetuam a classificação dos *patches*.

Para descrever o trabalho realizado, o restante desse artigo está organizado da seguinte maneira. A Seção 2 relata trabalhos relacionados. A Seção 3 detalha o *pipeline* de dados proposto. A Seção 4 apresenta um estudo para avaliação do *pipeline* através de experimentos, enquanto a Seção 5 traz as conclusões.

2. TRABALHOS RELACIONADOS

Os trabalhos relacionados à detecção de anomalias em vídeo podem ser caracterizados pela quantidade de cenas envolvidas, pela disponibilidade de dados durante treinamento e pelo nível de protocolo de avaliação.

Cenas Múltiplas A abordagem com cenas múltiplas emprega cenas capturadas a partir de vários pontos de vista. Sultani et al. (2018) propôs recentemente o *UCF-Crime*: um conjunto de dados com vídeos capturados de cenas múltiplas, obtidos na Internet. Dubey et al. (2019) conduziu experimentos no UCF-Crime com um extrator de características baseado em 3D ResNet (*3D Residual Networks*). Zhong et al. (2019) também realizou experimento no *UCF-Crime* aplicando rede convolucional em grafos (*graph convolutional networks-GCN*). No entanto, trabalhar com cenas múltiplas pode requerer conhecimento prévio dos parâmetros intrínsecos de cada câmera, bem como a atualização periódica de seus parâmetros extrínsecos. Nesse contexto, pode não ser trivial para um modelo único mapear, por exemplo, que uma área gramada é restrita apenas em determinada região da cena, ou que um veículo transitando representa normalidade apenas em um sentido da via.

Cena Única Uma formulação em cena única, por outro lado, não requer calibração de câmera ou informações adicionais e, portanto, é válida como uma abordagem inicial para estudos de detecção de anomalias em vídeo. Além disso, a detecção de anomalias em vídeos de cena única é uma área de pesquisa ainda em aberto (Liu et al., 2018, 2019; Zhou et al., 2019; Ravanbakhsh et al., 2019; Singh et al., 2020) e com diversas aplicações em potencial.

A disponibilidade de dados rotulados e eventos anormais durante o período de treinamento do modelo (Liu et al., 2019), é o critério utilizado para categorizar uma abordagem como não-supervisionada, semi-supervisionada ou supervisionada. Na detecção não-supervisionada, o modelo não tem acesso à classificação real das amostras. Na semi-supervisionada, somente dados normais são fornecidos. Na detecção supervisionada, dados rotulados normais e anormais são fornecidos durante treinamento. A detecção supervisionada pode ser dividida em duas sub-categorias: conjunto fechado (*closed-set*) e conjunto aberto (*open-set*). O critério, nesse caso, é a fração dos tipos de anomalias que serão disponibilizados para o modelo durante seu treinamento(Scheirer et al., 2012; Liu et al., 2019), ou seja:

- **Supervisionada em Conjunto Fechado**: todos os tipos de anomalias estão disponíveis ao modelo durante o treinamento.
- **Supervisionada em Conjunto Aberto**: disponibiliza apenas uma fração dos tipos de anomalias para o modelo durante seu treinamento.

Protocolo de avaliação em Nível de *Frame* O protocolo em nível de *frame* considera anômalo(positivo) o *frame* em que quaisquer um de seus pixels é inferido como anômalo, e todos os outros *frames* como normais(negativos). Este critério não considera a localização espacial, e classifica um *frame* como uma detecção correta (verdadeiro positivo), mesmo que os pixels anômalos detectados não estejam sobrepostos a quaisquer pixels anômalos das anotações *ground-truth* [3]. Mesmo os autores que propuseram este critério afirmaram não considerá-lo o melhor a ser usado(Li et al., 2013).

Protocolo de avaliação em Nível de *Patch* O protocolo a nível de *patch* considera a localização espacial dos eventos anômalos e, portanto, permite que verdadeiros positivos

[2] Clipe é um subconjunto dos *frames* de um vídeo

[3] O *ground truth* é um termo usado para se referir a informações que são conhecidas como reais ou verdadeiras, fornecidas por observação direta e medição (ou seja, evidências empíricas) em oposição a informações fornecidas por inferência. Nesse contexto, são os pixels que idealmente são considerados pertencentes a um evento anômalo.

e falsos positivos sejam corretamente classificados. Dessa forma, pode retornar resultados corretos mesmo quando um *frame* possui múltiplas anomalias, verdadeiros positivos simultâneos a falsos positivos, ou quando há múltiplas detecções de falsos positivos. Esse protocolo guarda semelhança com os critérios de detecção de objetos, ao usar a interseção sobre a união (*Intersection over Union*-IoU) entre uma região anômala *ground-truth* e uma região anômala inferida para determinar se uma anomalia foi detectada. Esse critério possibilita uma localização espacial aproximada (Ramachandra and Jones, 2020). No entanto, é importante salientar que as anotações *ground-truth* podem conter alguma imprecisão inerente ao processo de medição. Portanto, é desejável permitir uma margem de tolerância com relação às inferências, sem penalizar a avaliação do modelo.

Os trabalhos de Singh et al. (2020) e Liu et al. (2019) podem ser categorizados como detecção de anomalia supervisionada em conjunto aberto, com protocolo em nível de *frame*, em vídeos de cena única. Singh et al. (2020) propôs uma agregação de conjuntos(*Aggregation of Ensembles*-AoE), usando redes rede neurais convolucionais (*Convolutional Neural Networks*-CNNs) pré-treinadas e um conjunto de classificadores. Liu et al. (2019), por sua vez, propôs uma combinação de um extrator de características convolucional bidimensional (Conv2D) com um LSTM convolucional (*Convolucional* LSTM-ConvLSTM) e *triplet loss*.

A proposta de Zhou et al. (2019) se enquadra na categoria de detecção de anomalia supervisionada em conjunto aberto, com protocolo em nível de *patch*, em vídeos de cena única. Zhou et al. (2019) alcançou desempenho promissor com uma proposta baseada em três componentes: fusão de informações espaciais e temporais, extração de características e um bloco de codificação. O *Motion Fusion Block*(MFB) resume informações espaciais e temporais de objetos em movimento, compactando clipes de vídeo em uma única imagem. O *Feature Transfer Block*(FTB) emprega o conhecimento de outras tarefas/domínios relacionados para aumentar o aprendizado de características espaço-temporais e a extração dos resultados fornecidos pelo MFB. O bloco de codificação é um algoritmo iterativo adaptativo, como uma nova versão de uma rede *Long Short-Term Memory*(LSTM).

Durante esse trabalho adotou-se uma abordagem de detecção de anomalias supervisionada em conjunto aberto, com protocolo em nível de *patch*, em vídeos de cena única. Utilizar o protocolo a nível de *patch* permite analisar a localização espacial de anomalias, e classificar corretamente múltiplos eventos anômalos em um *frame*. Além disso, identificou-se que um *pipeline* de dados bem estruturado é essencial para o sucesso do modelo de detecção de anomalias em vídeo. Portanto, foram desenvolvidas várias etapas sequenciais que realizam o seccionamento dos vídeos em clipes, fusão da informação espaço-temporal, detecção de pontos de interesse e extração de *patches*, fornecendo-os para o treinamento e teste do modelo. Seguindo Zhou et al. (2019) o *pipeline* proposto dispõem de uma etapa de compactação (análoga ao MFB). O modelo utilizado para avaliar o *pipeline* possui um extrator de características *MobileNet* (Howard et al., 2017) pré-treinado no conjunto de dados *Imagenet* (Deng et al., 2009), seguido de um classificador,baseado em camadas totalmente conectadas, que retorna um *score* de pertencimento às classes normal e anômala. Os experimentos foram realizados com o *Street Scene* (Ramachandra and Jones, 2020), com avaliação através das métricas AUC e AUPRC, conforme detalhado na Seção 4.2.

3. PROPOSTA

No *pipeline* proposto, os dados de entrada são organizados como tuplas de observações e alvos. As observações são vídeos de entrada, que podem ser representados como uma sequência de *frames*:

$$\mathcal{X} = \{I_t, \cdots, I_{t-(T-1)}\}\ ,\ I_* \in \mathbb{R}^{H\times W\times C}, \tag{1}$$

onde T é quantidade de *frames* do vídeo, $H \times W$ são as dimensões dos *frames* e C é a quantidade de canais.

Os alvos, por sua vez, são rótulos [4] binários em nível de pixel atribuídos em cada *frame* de entrada:

$$\mathcal{Y} = \{l_t, \cdots, l_{t-(T-1)}\}\ ,\ l_* \in \{0,1\}^{H\times W}. \tag{2}$$

Dessa forma, o conjunto de dados de entrada é configurado em tuplas de observações e alvos, ou seja:

$$\mathcal{T} = (\mathcal{X}, \mathcal{Y}) \tag{3}$$

A saída do *pipeline* é um conjunto de *patches*, $\mathcal{P}$, extraído de $\mathcal{T}$, e definido como:

$$\mathcal{P} = \{\mathbf{x}_p\}_{p=1}^{P}, \mathbf{x}_* \in \mathbb{R}^{P\times S\times S}, \tag{4}$$

onde P é a quantidade de *patches* extraídos das observações, $\mathbf{x}_p$ é um *patch* e S é o tamanho do lado do *patch*.

O conjunto de *patches*, $\mathcal{P}$, é naturalmente desequilibrado, ou seja, a quantidade de *patches* da classe normal supera em muito a quantidade de *patches* anômalos. Para mitigar esse desequilíbrio, durante a construção dos conjuntos treino e teste para o modelo classificador, os *patches* anômalos são sobre-amostrados através de aumento de dados aleatórios. Portanto, o conjunto de *patches* para treino (x_{train}) e de teste (x_{test}) são definidos como:

$$x_{\text{train}} = \{a(\mathbf{x}_1, r), \cdots, a(\mathbf{x}_n, r)\}, \tag{5}$$

$$x_{\text{test}} = \{a(\mathbf{x}_{n+1}, r), \cdots, a(\mathbf{x}_{N+1}, r)\}, \tag{6}$$

onde $a(\bullet)$ é uma função de aumento de dados, que retorna uma quantidade r de transformações aleatórias de *patches* anômalos ou, em caso de *patches* normais, uma cópia inalterada do próprio *patch* de entrada.

Durante teste do modelo classificador, o conjunto de predições para um lote de b *patches* de teste, é dado por:

$$\hat{\mathcal{Y}} = \{\hat{y}_1, \cdots, \hat{y}_b\}\ ,\ \hat{y}_* \in \{0,1\}, \tag{7}$$

e é definido por:

$$\hat{\mathcal{Y}} = \{f(x_{\text{test}}[1]), \cdots, f(x_{\text{test}}[b])\}, \tag{8}$$

onde $f(\bullet)$ é um modelo do classificador que infere um rótulo $\hat{y}_i$ para cada patch $\mathbf{x}_i$ de entrada.

Em resumo, o *pipeline* proposto é ilustrado na Figura 1, e possui dois blocos principais, o primeiro é o extrator de *patches* e o segundo é o modelo classificador de *patches*. Ambos serão explicados em detalhe na Seção 3.1 e 3.2, respectivamente.

[4] Rótulo é a saída final esperada de um modelo de classificação. No caso em estudo, anômalo (1) ou normal (0).

$$\mathcal{T} \rightarrow \boxed{\text{extrator}} \rightarrow \mathcal{P} \rightarrow \boxed{\text{classificador}} \rightarrow \hat{\mathcal{Y}}$$

Figura 1. *pipeline* de dados: a sequência de tuplas de observações e alvos $\mathcal{T}$ é convertida em um conjunto de *patches* $\mathcal{P}$.

3.1 Extrator de patches

O Extrator é formado pelas etapas descritas a seguir.

Etapa 1. Janelamento Nesta etapa são extraídas tuplas de clipes e alvos com sobreposição temporal de passo unitário (Bento et al., 2020).

O conjunto de clipes

$$\mathcal{X}_t = \{X_T, \cdots, X_k\} \ , \ X_* \in \mathbb{R}^{Q \times H \times W \times C} \tag{9}$$

é definido como:

$$\mathcal{X}_t = \{I_t\}_{t=T-(Q-1)}^{T}, \cdots, \{I_t\}_{t=k-(Q-1)}^{k} \tag{10}$$

O conjunto de alvos

$$\mathcal{Y}_t = \{Y_T, \cdots, Y_k\} \ , \ Y_* \in \mathbb{B}^{Q \times H \times W} \tag{11}$$

é definido como

$$\mathcal{Y}_t = \{\{l_t\}_{t=T-(Q-1)}^{T}, \cdots, \{l_t\}_{t=k-(Q-1)}^{k}\} \tag{12}$$

Portanto, considerando $\mathcal{T}_w = (\mathcal{X}_t, \mathcal{Y}_t)$, a operação de janelamento, $w(\bullet)$, é definida como:

$$\mathcal{T}_w = w(\mathcal{T}), \tag{13}$$

Etapa 2. Compactação Essa operação resulta em uma representação espaço-temporal compactada de cada tupla de $\mathcal{T}_w$.

O conjunto de clipes compactados

$$\mathcal{U}_t = \{U_T, \cdots, U_k\} \ , \ U_* \in \mathbb{R}^{H \times W \times C}, \tag{14}$$

é definido por:

$$\mathcal{U}_t = \{m(X_T), \cdots, m(X_k)\}, \tag{15}$$

onde $m(\bullet)$ é um modelo que extrai características espaço-temporais dos clipes de entrada.

O conjunto de alvos compactados

$$\mathcal{V}_t = \{V_T, \cdots, V_k\} \ , \ V_* \in \{0,1\}^{H \times W}, \tag{16}$$

é definido por meio de conjunção lógica [5]:

$$\mathcal{V}_t = \left\{ \bigwedge_{i=T}^{T-(Q-1)} Y_T[i], \cdots, \bigwedge_{i=k}^{k-(Q-1)} Y_k[i] \right\}, \tag{17}$$

ou seja, o rótulo de um pixel $V_t[h,w] = 1$ se todos rótulos $Y_t[h,w,k]_{i=t-(Q-1)}^{t}$ forem iguais a 1, conforme ilustrado na Figura 2.

Portanto, considerando $\mathcal{T}_c = (\mathcal{U}_t, \mathcal{V}_t)$, a operação de compactação, $c(\bullet)$, é definida como:

$$\mathcal{T}_c = c(\mathcal{T}_w), \tag{18}$$

Etapa 3. Detecção de pontos de interesse O objetivo é detectar pontos notáveis (*keypoints*) nos clipes compactados, como cantos ou finais de segmento de linha.

O conjunto de pontos de interesse de um clipe compactado U_t, tem a seguinte forma

$$Kp_t = \{kp_1, \cdots, kp_n\} \ , \ kp_* \in \mathbb{R} \times \mathbb{R}, \tag{19}$$

[5] A conjunção é uma operação lógica representada pelo conectivo $\wedge$. O resultado dessa operação é verdadeiro se, e apenas se, todos seus operandos são verdadeiros

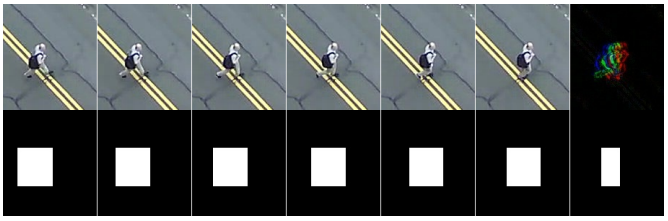

Figura 2. Exemplo para um instante t: clipe X_t (em cima), alvo Y_t(embaixo), clipe compactado U_t e alvo compactado V_t(ambos à direita).

e é definido por:

$$Kp_t = s(U_t), \tag{20}$$

onde $s(\bullet)$ é a função que detecta os pontos de interesse do clipe compactado. Portanto, o conjunto de todos pontos de interesse de um vídeo é definido como

$$\mathcal{K} = \{s(U_T), \cdots, s(U_k)\} \tag{21}$$

Etapa 4. Definição de uma grade de pontos Em torno de cada ponto de interesse kp_i é definida uma grade de pontos:

$$G_i = \{g_1, \cdots, g_{n_x \times n_y}\} \ , \ g_* \in \mathbb{R} \times \mathbb{R}, \tag{22}$$

a qual é parametrizada pela quantidade de pontos n_x ao longo do eixo x, a quantidade de pontos n_y ao longo do eixo y, pelo espaçamento horizontal gs_x e o espaçamento vertical gs_y (Figura 3).

Com isso, define-se o conjunto de todos os pontos de grade de um vídeo como

$$\mathcal{G} = G_T, \cdots, G_k \tag{23}$$

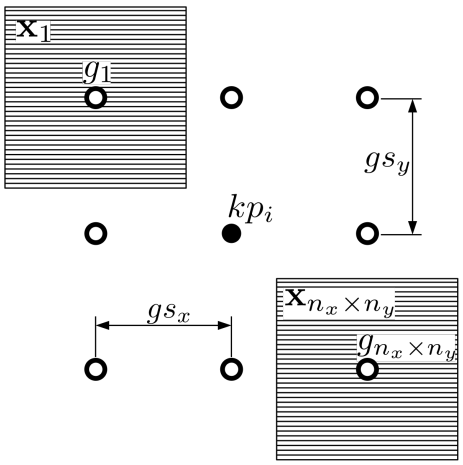

Figura 3. Grade de pontos em torno do ponto de interesse kp_i, com parâmetros $n_x = n_y = 3$ e espaçamento (gs_x, gs_y). Os patches $\mathbf{x}_1$ e $\mathbf{x}_{n_x \times n_y}$ foram construídos em torno dos respectivos pontos de grade g_1 e $g_{n_x \times n_y}$

Etapa 5. Criação e filtragem de *patches* Os pontos de grade são utilizados como referência para extração dos *patches*. No entanto, nem todos pontos de grade são mapeados a um *patch*, ou seja

$$\mathcal{P} = h(\mathcal{G}) \ , \ h : \mathcal{G} \nrightarrow \mathcal{P} \tag{24}$$

Isso ocorre pois é realizado um descarte de *patches* com base em sua redundância. *Patches* redundantes (Figura 4) são aqueles que possuem muitos pixels em comum com outros já existentes. A filtragem dos *patches* é realizada

mediante um limiar α de IoU, ou seja, da extensão da sobreposição entre *patches*(Figura 5). Também são descartados *patches* que possuem pixels fora da imagem.

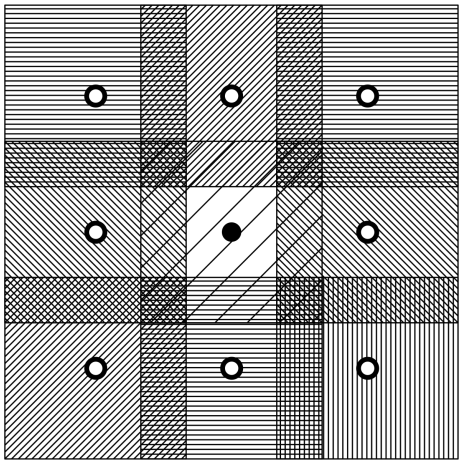

Figura 4. Patches redundantes.

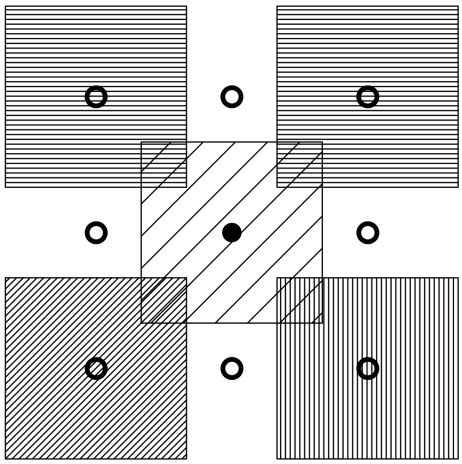

Figura 5. Patches remanescentes após filtragem.

Etapa 6. Marcação de *patches* anômalos Para rotular cada *patch* $\mathbf{x}_p$ do conjunto $\mathcal{P}$ (Equação (4)), pertencente a um clipe compactado U_t, é verificado um limiar β de IoU entre $\mathbf{x}_p$ e a matriz de anotação binária de alvos $\mathcal{V}_t$. Se o IoU for maior que β, o *patch* é marcado como anômalo, conforme ilustrado na Figura 6.

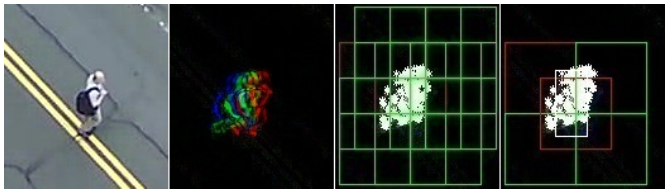

Figura 6. Da esquerda para a direita: conjunto de pixels do *frame* I_t do clipe X_t, clipe compactado U_t, pontos de interesse com patches redundantes, patches filtrados/rotulados(anômalo predito em vermelho e anotação em retângulo branco).

3.2 Modelo de Classificação de Patches

O modelo de classificação de *patches* é formado por um extrator de características e um classificador que retorna um *score* de pertencimento as classes normal e anômala. Na Figura 7 é mostrada a arquitetura do classificador.

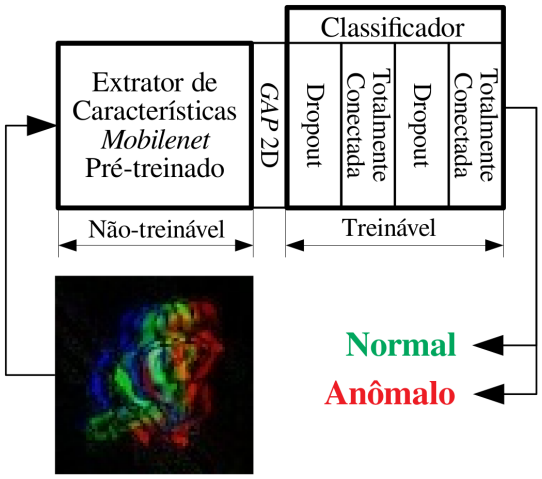

Figura 7. Arquitetura do classificador de *patches*.

O extrator de características consiste de um modelo *MobileNet* (Howard et al., 2017), pré-treinado no conjunto de dados *Imagenet* (Deng et al., 2009).

O classificador é constituído por duas camadas totalmente conectadas com 512 e 2 unidades, respectivamente; 2 camadas de *Dropout* , ambas com taxa de 50%, precedendo as camadas totalmente conectadas. Os hiperparâmetros do classificador foram definidos empiricamente.

4. EXPERIMENTOS

Nesta seção é descrito o conjunto de dados utilizado nos experimentos, algumas características relevantes de *software* e *hardware* utilizadas, e os resultados obtidos na avaliação do *pipeline* proposto.

4.1 Conjunto de Dados

Os experimentos foram realizados no conjunto de dados *Street Scene* (Ramachandra and Jones, 2020), o qual é focado em detecção de anomalias em vídeos de cena única. Consiste em 46 vídeos de treinamento e 35 vídeos de teste tirados de uma câmera USB estática, voltados para baixo, em cena com uma rua de faixa dupla, com ciclovias e calçadas de pedestres. Há um total de 203.257 *frames* de vídeo em cores (56.847 para treinamento e 146.410 para teste) com resolução de 1280 × 720. Os *frames* foram extraídos dos vídeos originais a 15 fps. São apresentados 17 tipos de eventos/atividades anômalas no conjunto de dados, como travessia de pedestre fora da faixa, carro estacionado ilegalmente, entre outros.

Os vídeos passam por uma preparação antes de serem entregues ao modelo de classificação de *patches*. Essa preparação é composta pelas etapas de preprocessamento de entrada, aumento de dados (*data augmentation*) e redimensionamento. Na etapa de preprocessamento de entrada,

os valores dos pixels são escalados entre -1 e 1, amostra por amostra. Esse procedimento prepara os dados para o extrator de características. A etapa de aumento de dados incrementa artificialmente a diversidade das amostras do conjunto de treino. Além disso, ajuda a expor o modelo a diferentes aspectos dos dados de treinamento e desacelera o sobre-ajuste (*overfitting*). Nessa etapa foram aplicadas as seguintes transformações aleatórias (mas realistas) durante o treinamento: inversão horizontal, inversão vertical e pequenas rotações (menores que $\frac{\pi}{5} rad$). O extrator de características *MobileNet* tem formato de entrada igual a 224×224 pixels, enquanto o tamanho dos *patches* de entrada é de 56×56 pixels. Por essa razão foi utilizada uma etapa de redimensionamento bidimensional (*2D upsampling*), com um fator de 4×4 para redimensionar os *patches* para 224×224 pixels.

4.2 Critério de Avaliação

O caso em estudo foi modelado como um problema de classificação multiclasse, com duas classes possíveis: normal ou anômalo. Por isso, as métricas fazem uso das seguintes definições básicas: VP são os verdadeiros positivos (*patches* corretamente classificados como anômalos), VN são os verdadeiros negativos (*patches* corretamente classificados como normais), FP são os falsos positivos (*patches* classificados erroneamente como anômalos), e FN são falsos negativos (*patches* classificados erroneamente como normais). Em detecção de anomalias, é prática usual avaliar o detector com base na curva ROC (*Receiver Operating Characteristic Curve*), a qual envolve a taxa de verdadeiros positivos (TVP) e a taxa de falsos positivos (TFP), onde: $TVP = \frac{VP}{VP+FN}$ e $TFP = \frac{FP}{FP+VN}$. Nesta curva, um detector de boa qualidade deve estar o mais próximo possível do canto superior esquerdo do gráfico TVP vs. TFP. Uma visualização alternativa é a curva *Precision* vs. *Recall* - PR. *Precision* indica qual o percentual dos itens selecionados é relevante (anômalo). *Recall*, por sua vez, representa qual percentual dos itens relevantes foi selecionado. Quantitativamente, $Precision = \frac{VP}{VP+FP}$ e $Recall = \frac{VP}{VP+FN}$. A interpretação da curva PR é um pouco diferente, pois um detector será tão melhor, quanto mais próxima sua curva estiver do canto superior direito. Nessa região da curva PR ocorre um melhor equilíbrio entre *Recall* e *Precision*. Para avaliar o desempenho do *pipeline* proposto, são utilizadas as áreas abaixo dessas curvas: AUC para a curva ROC, e AUPRC para a curva *Precision* vs. *Recall*.

Foi adotado um protocolo simplificado para avaliação de detecção de anomalias em nível de *patch*. Avaliou-se a atribuição de um rótulo de anormalidade a cada *patch* de teste, com base na probabilidade da classe anômala na saída do modelo, mediante a ultrapassagem de um limiar pré-definido. Este procedimento de avaliação é iterado usando uma faixa de limiares para construir as curvas ROC e *Precision* vs. *Recall*. O desempenho final do modelo é dado, portanto pela métricas AUC e AUPRC.

4.3 Implementação e Treino

Os *patches* foram extraídos com $S = 56$ (Equação (4)) e, portanto, com tamanho de 56×56 pixels. Durante a etapa de janelamento dos vídeos, os clipes foram construídos com comprimento Q de 6 *frames*. Na etapa de compactação foi utilizada a metodologia *Start-RGB* (dos Santos et al., 2020) para realizar a compactação dos clipes, a qual foi representada pelo modelo m na Equação (15). Durante a etapa de extração de pontos de interesse, a função s da Equação (21) foi implementada com o algoritmo *Surf* (*Speeded up robust features*) de Bay et al. (2006), disponível na biblioteca OpenCV (Bradski, 2000), versão 3.4.2. Durante a definição da grade de pontos, foram utilizados parâmetros $n_x = n_y = 3$, e $gs_x = gs_y = 28$ pixels. A marcação de *patches* anômalos foi condicionada a um limiar $\beta = 0,1$ de IoU. Para filtrar *patches* redundantes foi considerado um limiar $\alpha = 0,2$ de IoU.

O modelo foi implementado e treinado utilizando o *framework* Tensorflow (Abadi et al., 2015) na sua versão 2.4.1. O treinamento foi realizado com lotes de 1024 amostras, ao longo de cinquenta épocas com parada antecipada (*early stopping*) condicionada à melhoria da acurácia de validação e paciência de cinco épocas. Foi utilizado o otimizador Adam (Kingma and Ba, 2017), com decaimento de taxa de aprendizagem em curva de tempo inverso (*inverse time decay learning rate schedule*) e taxa de aprendizado inicial de 1×10^{-4}.

O *Street Scene Dataset* foi concebido por seus autores para aprendizado semi-supervisionado, portanto, seu conjunto de treino originalmente contém apenas *frames* sem anomalia. Isso ocorre possivelmente pois, na prática, ocorrem apenas alguns escassos eventos anômalos. No entanto, é plausível que mesmo uma fração dos raros eventos anômalos observados podem ajudar a detectar anomalias idênticas ou similares no futuro. Portanto, seguindo Liu et al. (2019), adotou-se a abordagem de detecção de anomalias supervisionada em conjunto aberto (*open-set supervised*), na qual apenas uma fração dos escassos eventos anômalos observados e numerosos eventos normais estão disponíveis durante o treinamento. Em resumo, o conjunto de treino que utiliza-se durante esse trabalho contém um fração dos tipos de anomalias da divisão original de teste, fornecida pelos autores do *Street Scene Dataset*. Todavia, não houve interseção entre conjuntos de vídeo fornecidos ao modelo durante treino e teste, evitando assim que o modelo avaliado observe os dados de teste, o que invalidaria a análise do desempenho. Dessa forma, os *patches* do conjunto de treino foram amostrados dos vídeos denominados "Test007", "Test011" e "Test017" no conjunto de dados original do *Street Scene* (Ramachandra and Jones, 2020), num total de 500.292 *patches* normais e 377.600 *patches* anômalos. Para os *patches* do conjunto de teste do modelo foram amostrados 468.927 *patches* normais e 468.927 anômalos, dos demais 32 vídeos da divisão de teste original dos autores do *Street Scene*, assegurando que não há interseção entre conjuntos de vídeos de treino e de teste na implementação. O conjunto de dados é naturalmente desequilibrado, ou seja, a quantidade de *patches* da classe normal supera em muito a quantidade de *patches* anômalos. Portanto, para mitigar esse desequilíbrio, a classe minoritária foi sobre-amostrada através de aumento de dados aleatório durante a construção dos referidos conjuntos treino e teste.

Os experimentos foram realizados em um computador com a seguinte configuração: (i) Sistema Operacional

Linux, distribuição Ubuntu 20.04.2, versão servidor; (*ii*) Processador Intel Xeon Silver 4214s(-HT-MCP-SMP-), 2.20GHz, com 48 núcleos; (*iii*) 64GB de RAM; (*iv*) 3TB de unidade de armazenamento permanente (disco rígido); (*v*) Placa gráfica *Nvidia Titan V*, com 12GB de memória dedicada.

4.4 Resultados

As Figuras 8 e 9 apresentam as curvas ROC e PR do modelo atendido pelo *pipeline* proposto. A partir da análise dessas curvas pode-se concluir que o modelo possui desempenho satisfatório nos conjunto de dados de teste. Sustenta tal constatação, o fato de que as curvas *ROC* e *Precision* vs. *Recall* mantiveram proximidade, respectivamente, ao canto superior esquerdo e direito ao longo de quase toda a faixa de iteração de limiares. Além disso, o modelo alcançou o considerável desempenho de AUC = 0.898 e AUPRC = 0.916.

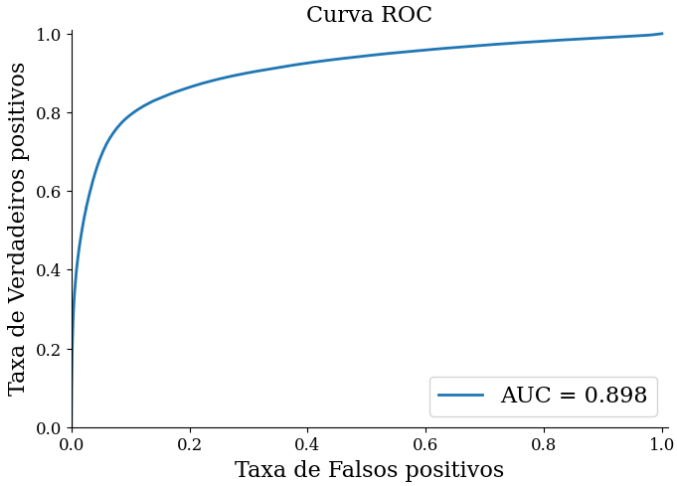

Figura 8. Curva ROC dos resultados do modelo.

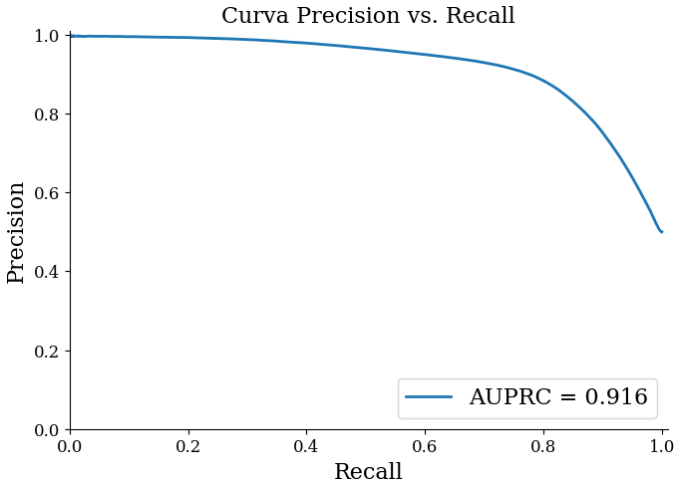

Figura 9. Curva Precision Recall dos resultados do modelo.

4.5 Discussão

O *pipeline* para extração de *patches* proposto oferece ao modelo de classificação *patches* com informações relevantes, pois são extraídos de pontos de interesse e incluem contexto espaço-temporal.

Observa-se, conforme ilustrado na Figura 10, que o modelo foi capaz de realizar a detecção de duas anomalias (em vermelho) simultâneas. No entanto, ocorreu um falso negativo (em amarelo) em evento com baixa variação espaço-temporal. Na Figura 11 também foi realizada mais uma detecção verdadeira positiva, no entanto o modelo retornou falso negativo para anomalia com pessoa parada na calçada (*loitering*). Os falsos negativos possivelmente ocorreram pois o modelo requer dados adicionais, referentes à posição absoluta de cada *patch* em seu *frame* correspondente.

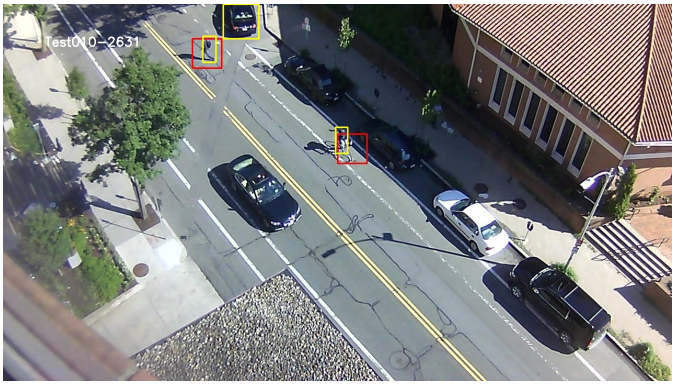

Figura 10. Duas detecções verdadeiras positivas, e um falso negativo no evento de baixa variação espaço-temporal: carro estacionado em local proibido (*car illegally parked*).

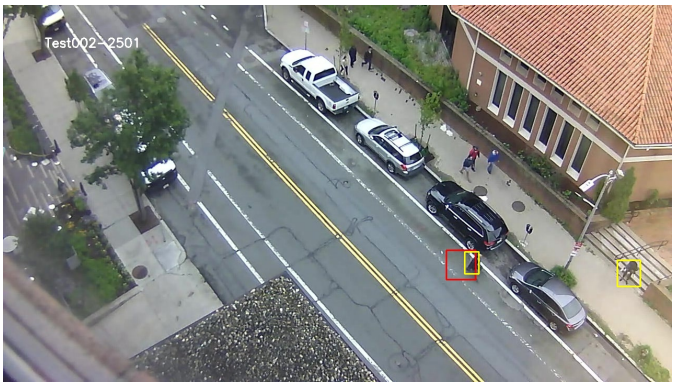

Figura 11. Dificuldade para detectar anomalias em eventos de baixa variação espaço-temporal, como a pessoa parada na calçada (*loitering*).

5. CONCLUSÃO

O monitoramento automático de eventos anômalos, através de vídeo, é uma área de pesquisa ainda em aberto e com diversas aplicações em potencial. Trabalhos recentes utilizam cenas múltiplas e protocolo em nível de *frame*. No entanto, detectar anomalias em vídeos de cenas múltiplas pode requerer conhecimento prévio e atualização periódica de parâmetros das câmeras. Além disso, o protocolo em nível de *frame* não observa a localização espacial dos eventos anômalos e pode conduzir a análises equivocadas. Nesse contexto, o presente trabalho tem foco no desenvolvimento de um *pipeline* detecção de anomalias supervisionado em conjunto aberto, operando ao nível de *patch*, em vídeos de cena única. Tal *pipeline* é constituído de extrator de dados que consiste do seccionamento dos vídeos em clipes,

fusão da informação espaço-temporal, detecção de pontos de interesse e extração *patches*. Essas etapas sequenciais extraem, preprocessam e transformam os dados de entrada em *patches*, para o treinamento e teste de um modelo de classificação neural de *patches*, orientado à detecção de anomalias. Tal modelo é composto por uma rede neural convolucional *MobileNet* pré-treinada e um classificador que retorna a classe estimada, normal e anômala.

Os experimentos foram realizados com o conjunto de dados *Street Scene*, alcançando AUC = 0.898 e AUPRC = 0.916. Tais resultados permitem concluir que: o extrator de dados oferece ao modelo *patches* com informações relevantes, pois são extraídos de pontos de interesse e incluem contexto espaço-temporal. Acredita-se que, para aprimorar a abordagem proposta em trabalhos futuros, durante o treinamento do modelo sejam utilizados dados adicionais da posição absoluta de cada *patch* em seu *frame* correspondente.

REFERÊNCIAS

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. URL `http://tensorflow.org/`. Software available from tensorflow.org.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. In *European conference on computer vision*, 404–417. Springer.

Bento, F.R.O., Vassallo, R.F., and Samatelo, J.L.A. (2020). Detecção de anomalias em vias públicas usando características espaciais e um classificador sequencial bidirecional. *Anais da Sociedade Brasileira de Automática*, 2(1).

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

dos Santos, C.C., Samatelo, J.L.A., and Vassallo, R.F. (2020). Dynamic gesture recognition by using cnns and star rgb: A temporal information condensation. *Neurocomputing*, 400, 238–254.

Dubey, S., Boragule, A., and Jeon, M. (2019). 3d resnet with ranking loss function for abnormal activity detection in videos. In *2019 International Conference on Control, Automation and Information Sciences (ICCAIS)*, 1–6. IEEE.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Kingma, D.P. and Ba, J. (2017). Adam: A method for stochastic optimization.

Li, W., Mahadevan, V., and Vasconcelos, N. (2013). Anomaly detection and localization in crowded scenes. *IEEE transactions on pattern analysis and machine intelligence*, 36(1), 18–32.

Liu, W., Luo, W., Li, Z., Zhao, P., Gao, S., et al. (2019). Margin learning embedded prediction for video anomaly detection with a few anomalies. In *IJCAI*, 3023–3030.

Liu, W., Luo, W., Lian, D., and Gao, S. (2018). Future frame prediction for anomaly detection–a new baseline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6536–6545.

Ramachandra, B. and Jones, M. (2020). Street scene: A new dataset and evaluation protocol for video anomaly detection. In *The IEEE Winter Conference on Applications of Computer Vision*, 2569–2578.

Ramachandra, B., Jones, M., and Vatsavai, R.R. (2020). A survey of single-scene video anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Ravanbakhsh, M., Sangineto, E., Nabi, M., and Sebe, N. (2019). Training adversarial discriminators for cross-channel abnormal event detection in crowds. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1896–1904. IEEE.

Santhosh, K.K., Dogra, D.P., and Roy, P.P. (2020). Anomaly detection in road traffic using visual surveillance: A survey. *ACM Computing Surveys (CSUR)*, 53(6), 1–26.

Scheirer, W.J., de Rezende Rocha, A., Sapkota, A., and Boult, T.E. (2012). Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7), 1757–1772.

Singh, K., Rajora, S., Vishwakarma, D.K., Tripathi, G., Kumar, S., and Walia, G.S. (2020). Crowd anomaly detection using aggregation of ensembles of fine-tuned convnets. *Neurocomputing*, 371, 188–198.

Sultani, W., Chen, C., and Shah, M. (2018). Real-world anomaly detection in surveillance videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6479–6488.

Yoshihashi, R., Shao, W., Kawakami, R., You, S., Iida, M., and Naemura, T. (2019). Classification-reconstruction learning for open-set recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4016–4025.

Zhong, J.X., Li, N., Kong, W., Liu, S., Li, T.H., and Li, G. (2019). Graph convolutional label noise cleaner: Train a plug-and-play action classifier for anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1237–1246.

Zhou, J.T., Du, J., Zhu, H., Peng, X., Liu, Y., and Goh, R.S.M. (2019). Anomalynet: An anomaly detection network for video surveillance. *IEEE Transactions on Information Forensics and Security*, 14(10), 2537–2550.

Zhou, J.T., Du, J., Zhu, H., Peng, X., Liu, Y., and Goh, R.S.M. (2019). Anomalynet: An anomaly detection network for video surveillance. *IEEE Transactions on Information Forensics and Security*, 14(10), 2537–2550.