UNIVERSIDADE FEDERAL RURAL DO RIO DE JANEIRO
INSTITUTO MULTIDISCIPLINAR

JOSÉ LUCAS ALVES GOMES

Classificação Automática de Anomalia em Vídeo Independente de Contexto

Prof. Filipe Braida do Carmo, D.Sc.
Orientador

Nova Iguaçu, Dezembro de 2021

Classificação Automática de Anomalia em Vídeo Independente de Contexto

José Lucas Alves Gomes

Projeto Final de Curso submetido ao Departamento de Ciência da Computação do Instituto Multidisciplinar da Universidade Federal Rural do Rio de Janeiro como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Apresentado por:

_____

José Lucas Alves Gomes

Aprovado por:

_____

Prof. Filipe Braida do Carmo, D.Sc.

_____

Prof. Leandro Guimarães Marques Alvim, D.Sc.

_____

Prof. Bruno José Dembogurski, D.Sc.

NOVA IGUAÇU, RJ - BRASIL

Dezembro de 2021

Agradecimentos

Gostaria de agradecer primeiramente a Deus por me permitir chegar até aqui.

Queria agradecer minha família pelo apoio e por me oferecer condições de continuar estudando e seguindo em frente.

Agradeço aos meus colegas e aos amigos que fiz nesses últimos anos. O caminho teria sido muito mais difícil se eu não tivesse tido a ajuda deles para enfrentar as dificuldades. Obrigado por todos os ótimos momentos.

Agradeço particularmente aos amigos do grupo que nós acabamos intitulando de Conselho. Graças a todos eles eu consegui me manter são e com esperança sempre.

E agradeço ao meu orientador, que aceitou me ajudar nesse trabalho, e me ofereceu uma ótima orientação. Definitivamente uma escolha que fico feliz em ter feito.

RESUMO

Classificação Automática de Anomalia em Vídeo Independente de Contexto

José Lucas Alves Gomes

Dezembro/2021

Orientador: Filipe Braida do Carmo, D.Sc.

Nos últimos anos, com o aumento da implantação de sistemas de câmeras de seguranças em espaços públicos, foi encontrada a necessidade de criar ferramentas que realizassem um tipo de análise automática nos vídeos capturados. Os ambientes capturados por essas câmeras por vezes possuem um comportamento esperado, e muitas situações que saem desse esperado não são bem-vindas, sendo necessário que hajam formas de identificar tal comportamento, chamado de anomalia, e amenizá-lo ou até mesmo preveni-lo. Anomalias são consideradas como qualquer comportamento que se destoe do padrão, e geralmente são indesejadas. A detecção de anomalias é uma área que se especializa na identificação automática de certo comportamento anômalo, e tal área tem sido assunto de um grande número de estudiosos de aprendizado de máquina, visão computacional, e muitas outras, e encontra aplicações em uma quantidade bastante ampla de outras áreas. As formulações da literatura, entretanto, costumam ser mais específicas a um certo problema, não havendo ainda muitas formas generalizadas de se aplicar detecção de anomalias. A proposta deste trabalho é, então, construir um modelo de detecção de anomalias que consiga discriminá-las sem conhecer seu contexto original. Será aplicado um modelo de reconstrução de imagens, com *Autoencoders* convolucionais, seguido da classificação das imagens em anomalias ou não.

ABSTRACT

Classificação Automática de Anomalia em Vídeo Independente de Contexto

José Lucas Alves Gomes

Dezembro/2021

Advisor: Filipe Braida do Carmo, D.Sc.

In recent years, with the increase of security camera systems implantations in public spaces, there has been found the necessity to create tools that would perform a kind of automatic analysis on the captured videos. The captured environments by these cameras sometimes have an expected behavior, and many situations that distance from this expected are not welcome, with the identification, softening, or even prevention of said behavior, known as anomaly, being a necessity. Anomalies are considered as any behavior that deviates from the pattern, and generally are unwanted. Anomaly detection is an area that specializes in the automatic identification of some anomalous action and said area has been the subject of a great number of scholars from machine learning, computer vision, and many other areas, and find applications in a very wide range of other areas. The literature formulations, however, tend to be more specific to a certain problem, not having yet many generalized ways to apply anomaly detection. The proposal of this work is, then, to build an anomaly detection model that can discriminate them without knowing their original context. An image reconstruction model, with convolutional autoencoders, will be applied, followed by the classification of images into anomalies or not.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ANN	*Artificial Neural Network*
MLP	*Multilayer Perceptron*
CNN	*Convolutional Neural Network*

Sumário

Capítulo 1

Introdução

Nos últimos anos, a implantação massiva de sistemas distribuídos de câmeras em espaços públicos aumentou a necessidade de ferramentas avançadas que operem a análise automática de fluxos de vídeos de segurança. Um desafio fundamental em segurança de vídeo inteligente é detectar eventos anômalos em cenas complexas e povoadas. Esse problema tem atraído atenção considerável da comunidade de pesquisa de visão computacional (XU et al., 2017).

Anomalias podem ser descritas em sua forma mais simples como atividades fora do padrão em um determinado contexto, *e.g.* anomalia em vídeos são quadros capturados que se destoam dos outros e não costumam ocorrer, como um carro andando em contra-mão, ou um assaltante em uma loja. Desta forma, é possível perceber que anomalias estão presente em quase tudo. A detecção de anomalias, entretanto, não é sempre uma tarefa trivial.

Detecção de anomalias tem sido o tópico de várias de *surveys* e artigos de revisão, bem como de livros (CHANDOLA; BANERJEE; KUMAR, 2009). Tanto o aumento do interesse no tópico quanto avanços na áreas de computação envolvidas em visão computacional, como *deep learning*, contribuiram para uma quantidade considerável de pesquisas sobre o assunto nos últimos anos.

Virtualmente todas as formas de detecção de anomalias tem duas premissas implícitas: (i) tem alguma forma de definir os padrões normais e (ii) divergências

da norma são indicadores de atividades indesejadas, *i.e.* anomalias. O conceito de detecção de anomalias é bastante intuitivo. Portanto, detecção de anomalias encontra suas aplicações em um vasto número de cenários diferentes (YAO et al., 2017).

Com a premissa de que anomalias são ocorrências malquistas, é possível notar porque esta área é de importante estudo. Idealmente, a identificação imediata de anomalias poderia amenizar cenários indesejados, como acidentes em auto-estradas e assaltos. Outro fator é que a detecção manual de anomalias é simplesmente inviável a partir de um certo ponto, principalmente devido ao já mencionado aumento do número de câmeras de vigilância.

No caso de detecção manual em tempo real, o processo irá requerer atenção constante do supervisor envolvido, o que em muitos casos é irrealista. Caso haja uma necessidade de revisar manualmente todos os dados envolvidos, *e.g.* uma sequência inteira de vídeos de segurança, na procura de anomalias, será preciso um trabalho demorado e tedioso de revisão, que continua não estando isento de falhas.

Consequentemente, a automação do processo de detecção de anomalia é interessante pois pode possivelmente diminuir a margem de erros, o que é crítico, e diminuir a carga humana para tal tarefa. A detecção automática, todavia, também encontra uma série de desafios e obstáculos. Um deles é que a formulação do problema tende a ser mais específica, fazendo que diferentes contextos requeiram diferentes abordagens.

Uma formulação genérica do problema implica, no âmbito computacional, em um modelo que seja capaz de discriminar anomalias independente do contexto original dos dados. Esse não é um desafio trivial, pois a forma como os dados são modelados nesse problema é de alta importância para o desempenho do mesmo, como comentado em Aggarwal (2017), além da possível disparidade entre diferentes cenários e ambientes.

A aplicação generalizada de técnicas de detecção de anomalia, porém, potencialmente oferece uma série de benefícios. A diminuição do esforço computacional decorrente do menor número de processos de detecção concorrentes, a facilidade de se construir modelos para contextos inéditos, e a maior adaptabilidade de cada modelo são alguns dos benefícios obtidos. Ambientes com diferentes câmeras e anomalias

distintas em cada uma podem se beneficiar bastante da aplicação de formulações mais genéricas do problema, ao invés de diversas formulações específicas.

1.1 Objetivo

O objetivo do trabalho é, então, construir um modelo de detecção automática de anomalias que seja independente do contexto dos dados. Será aplicado uma técnica de reconstrução de imagem em cada contexto utilizado para então construir um classificador que identifique as anomalias e classifique cada imagem em padrão ou anomalia. Espera-se que o classificador obtenha um bom resultado de acordo com as métricas utilizadas, independentemente da origem de cada imagem.

1.2 Organização do Trabalho

O trabalho está organizado da seguinte forma:

O capítulo 1 fez uma breve introdução ao tema e ao objetivo do trabalho.

O capítulo 2 apresentará uma base teórica para todos os conceitos utilizados ou relacionados ao longo do trabalho.

O capítulo 3 mostrará a proposta utilizada para resolver o problema exposto, tal como trabalhos relacionados e uma visão geral do modelo utilizado.

O capítulo 4 expõe a metodologia utilizada, tal como uma visão mais técnica do modelo, os experimentos realizados e seus resultados.

Por fim, o capítulo 5 contém a conclusão, com as considerações finais, limitações encontradas e possíveis trabalhos futuros.

Capítulo 2

Fundamentação

Nesse capítulo será apresentada uma base teórica necessária a fim de facilitar o entendimento dos temas abordados e das técnicas utilizadas no trabalho. Os temas estão organizados da seguinte forma: primeiro é apresentado o que é detecção de anomalias, tal como suas motivações, aplicações e desafios. Em seguida, são apresentados os temas de aprendizado de máquina e seus diferentes tipos. Por último, redes neurais, algumas variações e temas relacionados que são relevantes ao problema abordado.

2.1 Detecção de Anomalia

Comportamentos como carros andando em uma estrada ou pedestres andando por uma calçada são considerados o padrão para o ambiente, e normalmente não levantam suspeitas. A normalidade porém, é comprometida quando um carro é observado andando na contra mão, ou até mesmo em uma calçada. Tais comportamentos podem ser rotulados como anomalias. Anomalias são geralmente comportamento indesejado e, portanto, é importante tomar conhecimento de formas de remediar ou até mesmo prevenir suas ocorrências.

No contexto da ciência de dados, Aggarwal (2017) define anomalia, ou *outlier*, como um ponto nos dados que é significantemente diferente do restante. Anomalias

podem ser identificadas em diversos tipos de dados, como vídeos, textos, e imagens. No núcleo da detecção de anomalias está a estimação de densidade: dado um número de amostras de entrada, anomalias são aquelas que residem em áreas de baixa densidade de probabilidade (ZONG et al., 2018).

Na maioria das aplicações, os dados são criados por um ou mais processos geradores, que podem tanto refletir atividades nos sistema ou observações coletadas sobre entidades. Quando um processo gerador se comporta de forma incomum, isso resulta na criação de anomalias (AGGARWAL, 2017). A área focada em aplicar métodos para identificar anomalias é chamada de detecção de anomalias, e tem como objetivo analisar tal comportamento incomum, que pode conter informações relevantes sobre algum sistema ou ambiente.

Pela sua natureza, o conceito de anomalias é bastante amplo. Em qualquer ambiente onde seja possível definir um comportamento comum, é possível encontrar ocorrências de anomalias e, portanto, pode-se empregar técnicas de detecção em diversas áreas, tornando o estudo de tais técnicas altamente valioso. A área na inteligência artificial de aprendizado de máquina, em particular, apresenta algumas técnicas que serão abordadas eventualmente.

2.1.1 Motivações e Aplicações da Detecção de Anomalias

As motivações para se estudar técnicas de detecção de anomalias são tão amplas quanto suas áreas de aplicação. Hodge e Austin (2004) mostraram uma lista de aplicações que utilizam detecção de anomalias, algumas sendo: detecção de fraude, processamento de pedido de empréstimo, diagnóstico médico, análise de redes, pesquisas farmacêuticas, etc. Todas as áreas possuem uma motivação em comum, que é identificar comportamento usual e indesejado.

Uma aplicação da detecção de anomalias é em vídeos e imagens de câmeras de segurança. Tentativas de assalto, pedestres na estrada e acidentes de carro podem ser detectados como anomalia desta forma. A figura 2.1 mostra três exemplos de uma mesma câmera de segurança. Nesse exemplo, na primeira figura não há ocorrência de anomalia, por ser o ambiente normal, já a figura 2.1b mostra uma ocorrência

anômala, onde é possível ver um invasor no local, segurando um objeto suspeito. A terceira figura mostra o local já em estado crítico, com um incêndio ocorrendo.

(a) Imagem normal (b) Imagem com anomalia (c) Imagem com anomalia

Figura 2.1: Detecção de anomalia em incêndio culposo capturado por câmera de segurança. A figura 2.1b identifica uma anomalia no formato de um invasor no estabelecimento, a figura 2.1c captura um momento onde o incêndio está ocorrendo

A aplicação de detecção de anomalias, nesse caso, poderia identificar o invasor de prontidão, e o incêndio culposo poderia ser remediado, ou até mesmo prevenido. A detecção da infração também poderia ajudar na mobilização rápida para que o invasor pudesse ser capturado no momento da ação, antes que o mesmo conseguisse fugir. A diferença entre a rapidez de detecção nesse caso seria crucial para evitar que danos graves sejam causados, seja parando o incêndio antes de sua ocorrência, ou antes que sua escala se tornasse ainda maior.

Uma aplicação da detecção de anomalias no campo médico é dada em Homayouni et al. (2021). Eles apresentaram um modelo de detecção de anomalias em dados de pacientes com COVID-19, como tomografias, raios-x e etc. De acordo com eles, o valor desse trabalho está em automatizar o processo de detectar e explicar potenciais anomalias, o que permitirá que clínicos que tem conhecimento do domínio, mas não possuem habilidades de ciência de dados, avaliar o efeito do nível das anomalias e seriedade das anormalidades em suas pesquisas nos dados de COVID-19.

Ainda no domínio da medicina, métodos de detecção de anomalias podem ser utilizados para identificar características nos dados mais cedo do que seria manualmente. Chalapathy e Chawla (2019) mencionam a descoberta de novas doenças como consequência da detecção de anomalias. Nesse caso, as anomalias encontradas são dados novos, dos quais há pouca ou nenhuma informação, sendo importante que a detecção de tais dados possa auxiliar os envolvidos em extrair informações valiosas.

Wong et al. (2002) propuseram um modelo capaz de identificar surtos de doenças automaticamente, analisando dados de visitas médicas emergenciais a um número de hospitais em uma cidade.

Nos exemplos mostrados, um sistema de detecção de anomalia tornaria possível detectar ocorrências que talvez passassem despercebidas. Tendo em vista essa possibilidade, seria de interesse desenvolver sistemas de detecção que pudessem ser utilizados para evitar situações de difícil detecção, como uma invasão domiciliar de madrugada. A detecção de anomalias, desta forma, conseguiria identificar comportamento inesperado ou indesejado no ambiente, e possibilitar que os envolvidos responsáveis tenham uma reação rápida até mesmo para casos graves como os antes mencionados.

2.1.2 Desafios da Detecção de Anomalias

A área de detecção de anomalias, entretanto, pode vir a encontrar desafios, alguns deles sendo mais imprevisíveis do que parecem. Pela amplitude da área, a própria natureza das anomalias pode variar dependendo do contexto. Chandola, Banerjee e Kumar (2009) comenta que aplicar técnicas desenvolvidas em um domínio em outro não é trivial, pois algumas flutuações mínimas em certos domínios, como temperatura corporal em dados médios, podem ser consideradas anomalias, enquanto a mesma flutuação no domínio de mercado de ações pode ser considerado normal.

Separar anomalias de ocorrências comuns não é uma tarefa trivial em alguns casos. Em outros casos, é possível até mesmo que certos pontos anômalos no conjunto de dados não sejam de interesse para o objetivo principal do modelo, *e.g.* um animal avistado em uma câmera de segurança de um estacionamento. Podem também surgir novas ocorrências no contexto observado, as chamadas *novelties*, e sua detecção é uma área por si só.

Detecção de *novelties* é a identificação de novos ou despercebidos padrões nos dados (MILJKOVIć, 2010). As *novelties* detectadas não são consideradas dados anômalos; ao invés disso, elas são aplicadas ao modelo de dados regular (CHALAPATHY; CHAWLA, 2019). Chalapathy e Chawla (2019) também expõem que as

novelties detectadas podem ter um valor de limiar atrelado a elas, e que pontos afastados desse limiar são então, classificados como anomalias. O desafio se dá, então, em descobrir tal limiar e observar quais *novelties* são de fato anomalias ou não

Adicionalmente, anomalias derivadas de comportamento malicioso às vezes pode ser difícil de identificar. Muitas vezes os atores de tal comportamento se adaptam para fazer as observações anômalas parecerem normais, como mencionado em Chandola, Banerjee e Kumar (2009). Um exemplo de tal comportamento é a ocorrência de *shilling attacks*. Geralmente, um *shilling attack* é realizado pela injeção de perfis maliciosos em um sistema a fim de enganá-lo e atingir seu objetivo (BARBIERI et al., 2021). É um método geralmente usado contra sistemas de recomendação, cujos ataques buscam imitar os dados verdadeiros do sistema.

Há também a existência de ruído, que tende a ser similar às anomalias em si, mas não são, necessariamente, anomalias, e portanto são difíceis de identificar e remover. Ruído pode ser definido como um fenômeno nos dados que não é de interesse ao analista, mas age como um empecilho para a análise de dados (CHANDOLA; BANERJEE; KUMAR, 2009). A presença de ruído é indesejada pois adiciona mais uma camada de obstáculos na análise de dados, fazendo com que o modelo precise distinguir dados anômalos de ruídos, nos quais a análise tem pouco interesse.

Frénay e Verleysen (2014) mencionam que lidar com ruído de classe está diretamente relacionado com detecção de *outliers* e anomalias, pois na baixa probabilidade de existir ruído, instâncias rotuladas de forma errada podem ser consideradas como *outliers*, e o contrário também ocorre. Eles também afirmam que, devido a essa similaridade, técnicas de detecção de anomalias podem ser usadas em detecção de ruído. Contudo, Frénay e Verleysen (2014) alertam que instâncias mal rotuladas não são necessariamente anomalias ou *outliers*, por esses serem conceitos subjetivos, como mostrado em Collett e Lewis (1976).

Em vista desses desafios, o problema de detecção de anomalias, em sua forma mais geral, não é um problema fácil de resolver. De fato, a maioria das técnicas de detecção de anomalias existentes resolvem uma formulação específica do problema. A formulação é induzida por vários fatores como a natureza dos dados, disponibilidade

de dados rotulados, tipos de anomalias e serem detectadas e etc. (CHANDOLA; BANERJEE; KUMAR, 2009)

Finalmente, tendo em vista tudo que foi exposto, o estudo das técnicas de detecção de anomalias se torna muito importante. Técnicas que abrangem diversas áreas simultaneamente vem sido estudadas e não são uma impossibilidade. Para expor algumas das formas de resolver esse problema, será preciso introduzir alguns conceitos encontrados comumente na pesquisa de detecção de anomalias.

2.2 Aprendizado de Máquina

Aprendizado de Máquina é o sub-campo da Inteligência Artificial preocupado com o desenvolvimento de modelos computacionais de aprendizado. Aprendizado de máquina é inspirado pelo trabalho em diversas disciplinas: ciências cognitivas, ciência da computação, estatística, complexidade computacional, teoria da informação, teoria de controle, filosofia e biologia (YAO; LIU, 2014).

O Aprendizado de Máquina estuda técnicas de aprendizado de forma que uma máquina consiga aprender informações do ambiente ao seu redor, baseado em experiência prévia, tal qual quanto o aprendizado humano. Yao e Liu (2014) afirmam que, do ponto de vista biológico, aprendizado de máquina é o estudo de como criar computadores que vão aprender por experiência e modificar suas atividades baseada no aprendizado, ao invés de computadores tradicionais cujas atividades não mudam a não ser que o programador explicitamente as mude.

Na área computacional, Mitchell (1997) definem que um programa de computador é dito aprender de uma experiência E com respeito a uma classe de tarefas T e métricas de desempenho P, se seu desempenho nas tarefas em T, medido por P, melhora com a experiência E. Naqa e Murphy (2015) define um algoritmo de aprendizado de máquina como um processo que usa dados de entrada para atingir a tarefa desejada sem ser literalmente programado para produzir uma saída particular, ao invés disso se alterando ou adaptando sua arquitetura através de experiência (repetição), o que é chamado de treinamento.

Todo modelo de Aprendizado de Máquina utiliza alguma forma de treinamento para aumentar sua experiência. Uma forma comum de treinamento, como mostram Naqa e Murphy (2015), é apresentar amostras dos dados de entrada com seus respectivos resultados, o algoritmo então busca uma configuração ótima para que ele não só consiga produzir a saída desejada quando apresentado com os dados de entrada, mas que consiga generalizar para produzir o resultado desejado de novos, antes não vistos dados.

Quanto às possíveis aplicações do aprendizado de máquina, essas são extensas. A área vem tido um estudo extenso em praticamente qualquer dominio atualmente, e alguns até mesmo a sinalizam como o pilar que movimenta a área de *Big Data*, como mencionado em Naqa e Murphy (2015). De fato, com todas as aplicações listadas da área de detecção de anomalia, e por essa ser uma sub área do aprendizado de máquina, é possível perceber o quão extensas chegam suas aplicações.

2.2.1 Tipos de Aprendizado de Máquina

Os algoritmos de aprendizado de máquina operam em cima de três modelos de aprendizado: aprendizado supervisionado, não supervisionado, e de reforço. Há também um tipo de aprendizado chamado de semi-supervisionado, que funciona como um meio termo entre os dois primeiros. A escolha do tipo de aprendizado a ser aplicado em um modelo é crucial no desempenho do mesmo.

O modelo de aprendizado supervisionado assume a avaliabilidade de um professor ou supervisor que classifica os exemplos de treino em classes e utiliza a informação dos membros da classe de cada instância de treinamento, enquanto que o não supervisionado identifica informações dos padrões de classe de forma heurística e o aprendizado por reforço aprende através de interações de tentativa e erro com seu ambiente (SATHYA; ABRAHAM, 2013).

Enquanto que no supervisionado, o modelo tenta aprender a partir dos pares entrada e saída provisionados, o aprendizado não supervisionado utiliza modelos probabilísticos para categorizar a entrada, a partir de conhecimento adquirido internamente ao longo do treinamento, como mostram Yao e Liu (2014). Entre os

dois está o aprendizado semi supervisionado, que, como mencionado em Goldberg (2009), busca estudar como computadores e sistemas naturais aprendem na presença de ambos dados rotulados e não rotulados.

2.3 Redes Neurais

Um conceito prevalente nos modelos de aprendizado de máquina nos dias atuais é o de redes neurais artificiais. O estudo de redes neurais artificiais (*artificial neural networks*, ou ANNs) foi inspirado em parte pela observação que sistemas de aprendizado biológicos são construídos de redes bastante complexas de nêutrons interconectados. Em uma analogia ingênua, redes neurais artificiais são construídas a partir de um conjunto de unidades simples, densamente interconectadas, onde cada unidade pega um número de entrada real, que pode ser a saída de outras unidades, e produz um único valor de saída real,que pode se tornar a entrada de outras unidades (MITCHELL, 1997).

2.3.1 *Perceptron*

Um exemplo de sistema de redes neurais artificiais é baseado em uma unidade chamada de *perceptron*. Um *perceptron* toma um vetor de entradas de valor real, calcula uma combinação linear dessas entradas, então gera saída 1 se o resultado é maior do que algum limiar e -1 caso contrário. Mais precisamente, dadas entradas x_l, até x_n, a saída $o(x_1, ..., x_n)$ computada pelo perceptron é

$$o(x_1, ..., x_n) = \begin{cases} 1 & \text{se } w_o + w_l x_l + w_2 x_2 + ... + w_n x_n > 0 \\ -1 & \text{caso contrário} \end{cases} \tag{2.1}$$

na qual cada w_i é uma constante de valor real, ou peso, que determina a contribuição da entrada x_i à saída do *perceptron* (MITCHELL, 1997).

O *perceptron* foi inicialmente introduzido por Rosenblatt (1958). Contudo, o entusiasmo inicial sobre o perceptron foi amortecido peça observação feita por Minsky

e Papert (1969) de que a habilidade de classificação do *perceptron* está limitada a problemas linearmente separáveis, e não a problemas não lineares comuns como a simples lógica XOR. Um avanço foi alcançado em 1975 pelo desenvolvimento do *perceptron* multicamada (*multilayer perceptron*, ou MLP) por Werbos (1974).(NAQA; MURPHY, 2015)

O MLP é uma rede neural de múltiplas camadas, feita para eliminar a limitação do *perceptron* quando se trata de funções não lineares. A figura 2.2 apresenta a arquitetura de um MLP, com seus nós, ou neutrôns, interconectados. O MLP apresenta uma ou mais camadas ocultas, que ficam entre as camadas de entrada e saída.

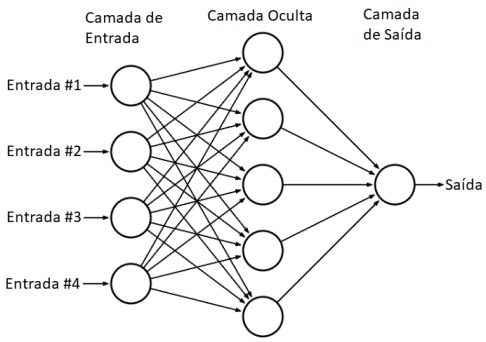

Figura 2.2: Arquitetura de um perceptron multicamadas. Imagem adaptada de Mohamed et al. (2015)

Os nós são conectados por pesos e sinais de saída que são funções da soma das entradas dos nós, modificados por uma simples transferência não linear, ou função de ativação. É a superposição de várias simples funções de transferência não linear que permitem o *perceptron* multicamadas a aproximar funções extremamente não lineares. Se a função de transferência fosse linear, então o MLP só conseguiria modelar funções lineares (GARDNER; DORLING, 1998).

2.3.2 *Backpropagation*

Treinar um MLP implica em alterar os pesos em cada nó até que a saída gerada se aproxime o mais possível da saída desejada. Assumindo que a saída desejada tem um valor x, e que todas as n combinações possíveis dos valores dos pesos de um MLP tenham uma distância d_n da saída desejada, o objetivo do treinamento é encontrar

a combinação de pesos com a menor distância d_n, ou menor erro, como mostrado em Gardner e Dorling (1998), sendo que o erro e a distância são proporcionais. O algoritmo de *backpropagation* é um algoritmo utilizado no treinamento para realizar tal otimização.

O algoritmo de *backpropagation* utiliza o gradiente descendente na tentativa de minimizar o erro quadrado entre o valor de saída da rede e o valor alvo da saída (MITCHELL, 1997). Gradiente descendente é uma forma de minimizar uma função objetivo $J(\Theta)$ parametrizada por parâmetros $\Theta \in \mathbb{R}^d$ atualizando os parâmetros na direção oposta do gradiente da função objetivo $\nabla_\Theta J(\Theta)$ no que diz respeito aos parâmetros (RUDER, 2017). No caso de um MLP, a função objetivo a ser diminuída são os erros, e os parâmetros, os pesos. Mitchell (1997) comenta sobre alguns problemas do gradiente descendente, como o dos mínimos locais, e que mesmo com esses problemas, o *backpropagation* gera excelentes resultados em várias aplicações do mundo real, na prática.

2.3.3 Função de Ativação

Funções de ativação são especialmente usadas em redes neurais artificiais para transformar um sinal de entrada em um sinal de saída, o que por sua vez é dado como entrada para a próxima camada na pilha. Numa rede neural artificial, é calculada a soma do produto das entradas com seus correspondentes pesos e finalmente aplicada a função de ativação ao valor para conseguir a saída daquela camada em particular, e fornecê-la como entrada da próxima camada (SHARMA; SHARMA; ATHAIYA, 2020).

Existem várias funções de ativações na literatura e estudá-las é importante quando se deseja construir um modelo de rede neural, já que o desempenho do modelo pode variar bastante dependendo da função escolhida. Por exemplo, pelo fato do *backpropagation* utilizar gradiente descendente, ele só funciona com funções diferenciáveis.

Sharma, Sharma e Athaiya (2020) mencionam que funções não-lineares são desejadas ao invés de funções lineares, pelo fato da maioria dos problemas do mundo

real possuírem características não lineares. Esse também é o motivo pelo qual funções de ativação são necessárias, já que sem elas, as redes neurais teriam sempre saídas lineares, e agiriam apenas como modelos de regressão linear com desempenho limitado.

Alguns exemplos de funções de ativação mais utilizadas são a sigmoide, na forma $f(x) = 1/e^{-x}$, que transforma a saída em um valor no intervalo de 0 a 1, a Tanh (tangente hiperbólica), na forma $f(x) = 2 * sigmoide(2x) - 1$, cuja saída é num intervalo de -1 a 1 e tem a vantagem de possuir valores de ambos sinais. A função Tanh é preferível ao invés da sigmoide por possuir gradientes que não estão restritos a variação em uma certa direção, e também por ser centralizada no zero (SHARMA; SHARMA; ATHAIYA, 2020).

Também há a função ReLU, que é amplamente usada em redes neurais. ReLU vem de unidade linear retificada e é definida como $f(x) = max(0, x)$. Como a ReLU elimina valores negativos, ela pode descartar informações importantes que poderiam estar contidos nesses valores. Além disso, quando a entrada é negativa, o gradiente será 0. Então, os pesos da rede neural não podem ser atualizados e continuam em silêncio durante o resto do processo de treinamento. A velocidade do aprendizado da rede fica devagar, e alguns neurônios podem se tornar ineficazes (YOU et al., 2020).

A função *LeakyReLU* foi criada, então, para resolver esse problema. A função tem a forma $f(x) = max(0.01x, x)$, e é possível perceber que ela permite valores negativos, portanto aliviando o problema do gradiente zero. Uma outra variação da ReLU é a ReLU Parametrizada, que introduz um parâmetro novo a na parte negativa da função. Ela é descrita na forma $f(x) = max(\alpha x, x)$. O valor de α, quando colocado em 0.01, a faz se comportar como a *LeakyReLU*, mas aqui α também é um parâmetro treinável. Para convergência ótima e mais rápida, a rede aprende o valor de α (SHARMA; SHARMA; ATHAIYA, 2020).

2.3.4 Redes Neurais Convolucionais

As redes neurais convolucionais, CNN, são um modelo típico de *deep learning* que tem fortes capacidades de extrair elementos automaticamente de dados puros

(YOU et al., 2020). Pelas suas características, as redes neurais convolucionais são comumente utilizadas para processar dados em forma de imagens. Com o avanço das GPUs e sua capacidade de processamento, as CNNs vêm ganhando ainda mais força para realizar tarefas mais complexas.

Redes neurais convolucionais tem tido resultados incríveis ao longo da última década em uma variedade de campos relacionados a reconhecimento de padrões; de processamento de imagem a reconhecimento de voz. O aspecto mais beneficial das CNNs é a redução do número de parâmetros em uma rede neural artificial. Essa conquista motivou ambos pesquisadores e desenvolvedores a alcançar modelos mais largos em vista de solucionar tarefas complexas, que não eram possíveis com ANNs clássicas (ALBAWI; MOHAMMED; AL-ZAWI, 2018).

Em uma ANN que visa processar uma imagem, o procedimento pode ser muito custoso. O número de conexões necessárias e pesos utilizados é simplesmente inviável em alguns exemplos. A técnica que define a viabilidade das CNNs é a chamada convolução. Na convolução, os neurônios não processam imagens inteiras, mas sim fragmentos da imagens, e camadas ocultas só processam os fragmentos recebidos das camadas anteriores. Albawi, Mohammed e Al-Zawi (2018) mostram um exemplo onde o número de conexões numa rede diminui de 3 milhões para 75 conexões, quando aplicada uma camada de convolução.

A figura 2.3 mostra um exemplo de convolução. Nela, pega-se algum fragmento da entrada e é aplicado um filtro, que determina como será a saída. Os filtros são bastante importantes nas camadas convolucionais e cada camada pode possuir um filtro diferente a fim de extrair informações diferentes. A quantidade de passos que um filtro dá em uma imagem a fim de ser aplicado em um número de fragmentos é chamada de *stride*. *Padding* é uma técnica utilizada para diminuir a perda de informação na borda da imagem, sendo o *zero padding* a forma mais utilizada. Nela, uma quantidade de novas bordas é adicionada na entrada, contendo apenas zeros.

Outro conceito importante utilizado nas CNNs é o de *pooling*. A ideia principal do *pooling* é diminuir a complexidade da entrada de algumas camadas. No processamento de imagens *pooling* pode ser considerado similar a diminuir a resolução. *Max-pooling*

é um dos tipos mais comuns de métodos de *pooling*. Ele particiona a imagem em retângulos de sub-região, e só retorna o valor máximo dentro daquela sub-região. Um dos tamanhos mais usados no *max-pooling* é o 2×2 (ALBAWI; MOHAMMED; AL-ZAWI, 2018). A figura 2.4 mostra um exemplo de *max-pooling* 2×2.

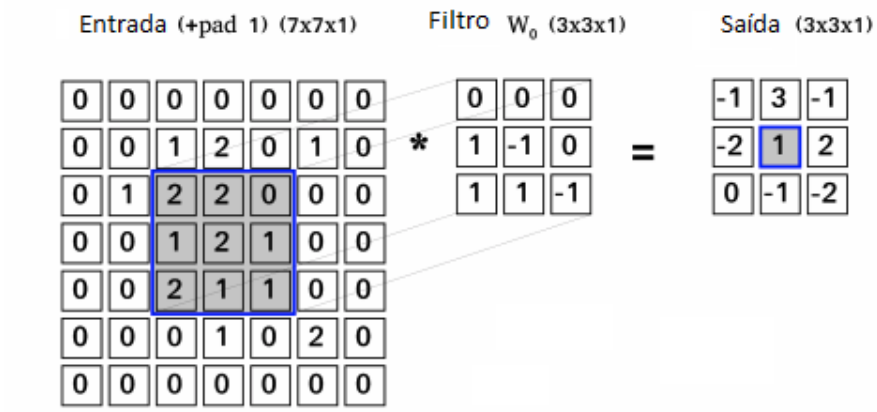

Figura 2.3: Exemplo de aplicação de uma convolução. A entrada é uma matriz 7×7 com uma camada de *padding*. Imagem adaptada do site *Cosmos*[1]

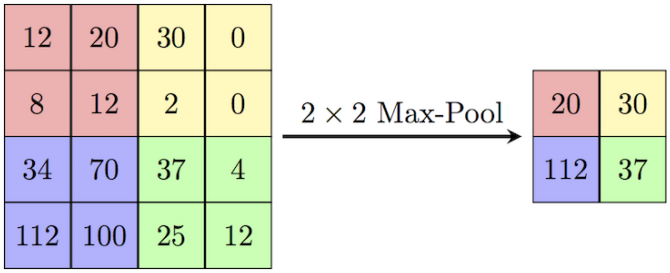

Figura 2.4: Exemplo de um *max-pooling* 2×2. Imagem retirada do site *Papers With Code*[2]

Há vários estudos comprovando a eficácia das CNNs. Krizhevsky, Sutskever e Hinton (2017) mostraram que uma grande rede neural convolucional profunda é capaz de alcançar resultados que superam os já existentes em um conjunto de dados altamente desafiador, utilizando puramente aprendizado supervisionado. Albawi, Mohammed e Al-Zawi (2018) comentam que um dos principais motivos é o fato dos problemas resolvidos por CNNs não precisam necessariamente ter elementos

[1] https://www.cosmos.esa.int/web/machine-learning-group/convolutional-neural-networks-introduction

[2] https://paperswithcode.com/method/max-pooling

espacialmente dependentes, *e.g* em um problema de detecção facial, não é preciso prestar atenção onde as faces estão localizadas na imagem, e sim tentar localizá-las independentemente da posição.

2.3.5 *Autoencoder*

Autoencoders são um tipo de rede neural não-supervisionada nas quais a saída desejada é igual a entrada. Em particular, *autoencoders* aprendem um mapeamento da entrada para si mesma através de um par de fases de codificação e decodificação $X' = D(E(X))$ onde X é o dado de entrada, E é o mapeamento de codificação do dado de entrada para as camadas ocultas, D é o mapeamento de decodificação das camadas ocultas para a camada de saída, e X' é a versão recuperada dos dados de entrada. A ideia é treinar E e D a fim de minimizar a diferença entre X e X' (ZHOU; PAFFENROTH, 2017)

O papel do *Autoencoder* não é tão trivial pois envolve o trabalho conjunto de um codificador, que transforma os dados de entrada, e o decodificador, que faz o processo reverso, como mostra a figura 2.5. As aplicações do *autoencoder* incluem remoção de ruído, redução de dimensionalidade, extração de atributos, dentre outras. Suas aplicações tornam o *autoencoder* uma ferramenta a ser considerada na área de detecção de anomalias, motivando uma série de estudos e experimentos na literatura utilizando essa ferramenta.

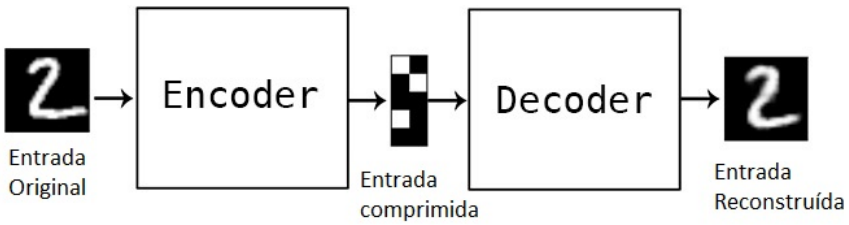

Figura 2.5: Arquitetura simplificada de um *Autoencoder*. Imagem adaptada do site *The Keras Blog*[3]

Uma extensão do *autoencoder* é o chamado *Autoencoder* convolucional. O *autoencoder* convolucional é extendido do *autoencoder* instanciando funções de codificação

[3]https://blog.keras.io/building-autoencoders-in-keras.html

e decodificação com redes neurais convolucionais (ZHAI et al., 2019). Ele possui as mesmas vantagens de uma CNN normal, simplificando o processamento da rede no processamento de imagens.

Capítulo 3

Proposta

Neste capítulo será exposto como foi feita a modelagem do problema de detecção de anomalia, tal como a motivação para o trabalho. Serão também comentados alguns trabalhos relacionados, com temas similares, e uma visão do modelo utilizado.

3.1 Introdução

Alguns ambientes possuem uma tolerância mais baixa à ocorrência de anomalias. Um assalto a uma loja ou um acidente de carro são situações críticas e detectá-las com prontidão deve possuir alta prioridade para a segurança do ambiente. Os responsáveis pela vigilância de um local podem considerar aplicar um sistema de detecção de anomalias em suas câmeras. Os mesmo sistemas podem permitir que diagnósticos médicos de hospitais tenham suas anomalias identificadas de antemão, e tentativas de fraude sejam prevenidas. Devido a isso, não é surpresa que a área de detecção de anomalias é de extremo interesse para a sociedade.

A detecção manual de anomalias, embora precisa, nem sempre é a mais conveniente. Humanos estão sujeitos a falhas e o trabalho de monitorar, por exemplo, câmeras de segurança se torna extremamente cansativo com o tempo. Múltiplas câmeras exigem ainda mais uma capacidade de multitarefa dos supervisores envolvidos. Em algumas aplicações, como anomalias em grande volumes de dados, é simplesmente irrealista

esperar alguma forma de detecção manual. Por isso é interessante estudar técnicas automáticas para detectá-las.

Ao longo desse trabalho, é possível perceber a amplitude das aplicações e definições de anomalias. Elas estão presentes em praticamente qualquer contexto e em muitas vezes não são tão claras, principalmente quando se trata de detecção automática via modelos em aprendizado de máquina. O que é anomalia em um dado contexto também muda com o tempo: é possível que comportamentos anormais acabem se tornando comum com o passar do tempo.

Essa amplitude ressalta um desafio recorrente na área: a generalização. Generalizar as ferramentas de detecção de anomalias possibilita resultados interessantes o suficiente para justificar enfrentar tais desafios. Por exemplo, aplicar um sistema único de detecção de anomalias em todas as câmeras de um estabelecimento pode tornar o trabalho menos computacionalmente intensivo do que aplicar um sistema diferente para cada câmera no local, e possivelmente mais barato.

Por estes e outros motivos, é importante estudar técnicas para generalizar a detecção automática de anomalias de forma eficaz. Esse problema, todavia, é mais complexo do que parece. Chandola, Banerjee e Kumar (2009), expõem alguns dos problemas relacionados à generalização de métodos de detecção de anomalias, como a avaliabilidade de dados rotulados em certas formulações do problema, os tipos de anomalias a serem detectadas e etc. De fato, as anomalias encontradas em ambientes como salas de aulas e uma auto-estrada são substancialmente diferentes.

O que todos os domínios tem em comum, entretanto, é a presença de anomalias. Se existem dados que se diferem do padrão o suficiente para aplicar técnicas específicas de detecção, uma característica comum a todos os dados anômalos é a diferença entre eles e os dados considerados normais. A proposta é, então, construir um modelo de detecção automática de anomalia que seja independente do domínio a ser aplicado, utilizando essa diferença de um dado anômalo a seu contexto normal, sem realmente precisar conferir o contexto original de cada dado.

3.2 Trabalhos Relacionados

Com o crescimento da área e seus desafios, foram feitos inúmeros trabalhos sobre detecção de anomalias ao longo do tempo. Hodge e Austin (2004) proveram uma *survey* com técnicas desenvolvidas para detecção de anomalia em aprendizado de máquina e estatística, que continua sendo relevante. Outra *survey* informativa é a de Chandola, Banerjee e Kumar (2009), abordando diversas técnicas e aplicações.

Na área de aprendizado não supervisionado, onde a base não precisa possuir rótulos, há uma série de trabalhos com abordagens diferentes. Zhou e Paffenroth (2017) expuseram uma técnica que usa dois *autoencoders* para não só descobrir *features* não lineares de alta qualidade, mas também conseguir eliminar anomalias e ruído sem nenhum acesso a qualquer conjunto de treinamento limpo. Zong et al. (2018) apresentaram um modelo de reconstrução em autoencoder junto com um Modelo de Mistura Gaussiana (*Gaussian Mixture Model*, ou GMM).

Xu et al. (2015) propuseram um modelo para aprender padrões em imagens num modelo espaço-temporal, e proclamam terem sido os primeiros a introduzir um *framework* de *deep learning* não supervisionado para construir automaticamente representações discriminativas para detecção de anomalia em vídeo. Esse trabalho também faz uso de *Autoencoders* para identificar os padrões considerados normais no conjunto de dados.

Schlegl et al. (2019) criaram um modelo de detecção de anomalia não-supervisionado utilizando GANs (*Generative Adversarial Networks*), chamado de *Fast AnoGAN*, que foi aplicado em dados da área da saúde. Yao et al. (2017) apresentam uma abordagem na detecção de anomalia como um serviço, enfatizando desafios e necessidades para sua implementação e democratização, na qual clientes podem fazer uso de tal serviço sem se preocupar com os detalhes.

Barros e Neto (2020) utilizaram um modelo de reconstrução de imagens via *autoencoder*, seguido de um classificador, que foi aplicado a um conjunto de dados a fim de classificar corretamente suas anomalias. Em seu trabalho foi aplicada uma técnica de extração do erro que representa a diferença entre a imagem reconstruída e

a original, e sua proposta é baseada nos mesmos princípios e intuições que a deste trabalho, que será exposta a seguir.

3.3 Proposta

Como mencionado, a proposta se baseia em construir um modelo capaz de detectar anomalias que seja independente de uma formulação específica. Fazendo isso, espera-se que seja possível identificar anomalias apenas baseado no fato das mesmas serem ocorrências de baixa densidade, ou seja, que são suficientemente diferente do normal. O modelo será utilizado em imagens que representam *frames* de vídeos cujas anomalias se quer identificar. Até certo ponto, é necessário que haja um tratamento nos dados desse vídeos para que estejam de uma forma mais geral.

Primeiro, será aplicado uma técnica de reconstrução de imagens não supervisionada em cada câmera, sendo cada uma representada por um conjunto de dados em vídeo diferente, com o objetivo de extrair informações que possam vir a denunciar a existência de anomalias em determinado *frame*. Na próxima etapa, o modelo então visa inferir, a partir dessas informações, se a imagem de onde elas se originam é uma anomalia ou não. O objetivo nessa etapa é que seja possível classificar anomalias sem conhecimento prévio do contexto ou origem das mesmas.

O modelo proposto foi organizado da forma como segue na figura 3.1. Foram utilizados um reconstrutor de imagens e um classificador, que serão expostos com mais detalhes a seguir. Em resumo, o reconstrutor recebe os dados, nesse caso, imagens, e tenta aprender a replicá-las. Então é calculada uma imagem de erro dada como a diferença entre a imagem original e a reconstruída, que o classificador tentará prever se é uma anomalia ou não.

O fluxo dos dados é o seguinte: primeiro eles são organizados e preparados para o reconstrutor. Cada *dataset* possui um reconstrutor diferente que aprenderá a replicar os padrões que aparecem nas ocorrências normais de cada situação. Cálculos feitos geram um conjunto novo de dados, chamado de conjunto de erros. Daí, os conjuntos de erros gerado por cada reconstrutor são agregados e entregues ao classificador.

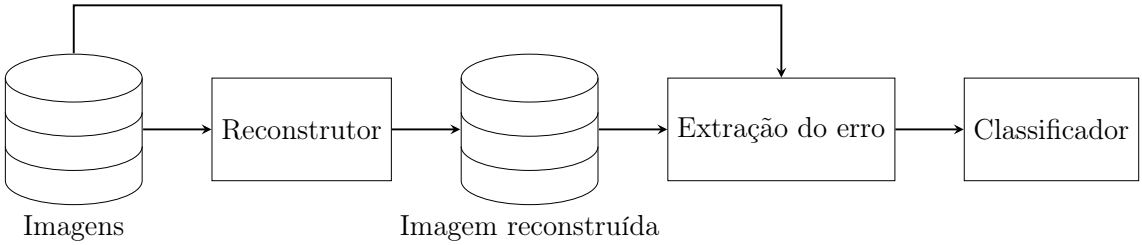

Figura 3.1: Fluxo do modelo proposto

A importância dessa técnica é que, dessa forma, o classificador não conhece, e nem precisa conhecer, nenhuma das imagens dos *datasets* originais. É esperado que, com os reconstrutor, sejam geradas imagens de erro boas o suficiente para que o classificador aprenda a prever se é anomalia ou não apenas com essas imagens, conseguindo assim uma forma de generalizar seu trabalho para inúmeros outros conjuntos de dados.

3.3.1 Reconstrução de Imagem

Primeiro, é necessário que haja uma forma de reconstruir as imagens que serão analisadas para a detecção de anomalias. A técnica utilizada envolve empregar um método de reconstrução de imagens a fim de aprender as *features* que compõem uma imagem normal no devido contexto, enquanto que as ocorrências anômalas são evitadas. Isso abre oportunidades que serão esclarecidas a seguir.

O método de reconstrução de imagem utilizada foi do *autoencoder* convolucional. Foram utilizadas as imagens que representam os *frames* não anômalos a fim de que esse aprendesse a reproduzir ocorrências normais em cada conjunto de dados. Dessa forma, é esperado que o *autoencoder* não consiga reproduzir bem certos acontecimentos em algumas imagens, e que tais acontecimentos sejam, na verdade, as anomalias que deseja-se identificar.

Nas figuras 3.2b e 3.2d, é possível perceber a ação do *autoencoder*. No caso da figura sem anomalia, ele aprendeu a reconstruir a imagem com certa fidelidade, e espera-se que a diferença entre a original e a reconstruída seja mínima. Já no caso com anomalia, pode-se perceber que o *autoencoder* encontrou problemas para reproduzir a ocorrências anômalas, no caso o carro e a bicicleta, é possível observar a

diferença do brilho do carro entre as duas imagens, o que é tanto o esperado quando desejado, para que a diferença entre as duas imagens, nesse caso, seja grande o suficiente para ser perceptível.

(a) Não anomalia

(b) Não anomalia reconstruída

(c) Anomalia

(d) Anomalia reconstruída

Figura 3.2: Anomalias e não anomalias e suas respectivas reconstruções

Como toda imagem pode ser representada por um conjunto de *pixels* em um computador, é calculada a diferença, *pixel* por *pixel* de todas a imagens em suas duas versões, e gerado um novo conjunto de imagens de erro. A figura 3.3 mostra uma destas imagens, com suas cores invertidas para melhor visualização, gerada pela diferença entre as figuras em 3.2c e 3.2d. Assumindo no exemplo que as áreas pretas signifiquem erro, é esperado que uma não anomalia apresente um resultado completamente branco, ou com partes pretas de pouca densidade. No exemplo é possível enxergar pontos pretos com alta densidade nas áreas anômalas presentes na figura 3.2c.

3.3.2 Classificador

As imagens de erro são cruciais para o prosseguimento do modelo. Serão elas as utilizadas para classificar uma imagem em anomalia ou não. Para isso será feito o

Figura 3.3: Exemplo de imagem de erro (cores invertidas)

uso de um classificador, que estará encarregado de aprender quais dessas imagens de erro são de fato anomalias.

Um fator importante que já foi comentado é o fato do classificador utilizar imagens de erro, e não as imagens originais. Essa técnica vai abrir espaço para que o classificador consiga aprender a rotular anomalias sem precisar conferir o contexto original da imagem, e que ele possa trabalhar com erros de diferentes conjuntos de dados simultaneamente.

As imagens precisam ser, a priori, rotuladas em anomalias ou não. Nessa etapa também é feita a junção de todas as imagens de diferentes conjuntos de dados em uma só base, que será dividida posteriormente entre treino e teste. Ao fim é esperado que o classificador tenha aprendido a identificar as imagens de erro que correspondem à anomalias. Em seguida, com o classificador pronto, é analisado o seu resultado com alguns experimentos que serão descritos a seguir.

Capítulo 4

Experimentos

Nesse capítulo serão expostas as técnicas utilizadas para resolver a proposta do capítulo 3. Primeiro serão apresentados os objetivos e metodologia empregada, as métricas utilizadas para avaliar o desempenho do modelo em cada experimento, e as bases de dados nas quais o modelo foi aplicado. Por fim, serão mostrados os experimentos realizados, e a análise dos resultados obtidos.

4.1 Objetivos e Metodologia

O objetivo do experimento é avaliar o modelo proposto por esse trabalho e verificar a capacidade de generalização deste. Desta forma, o classificador deverá ser capaz de identificar anomalias utilizando as imagens reconstruídas pelos *autoencoders* independentemente do ambiente realizado para a gravação da imagem. Complementando, será necessário treinar esses *autoencoders* para cada ambiente utilizado para avaliação. Tendo em vista esse cenário, serão necessários dois experimentos: primeiro para os *autoencoders* e o segundo para o classificador de anomalia.

O primeiro experimento será responsável pela criação dos *autoencoders* para cada câmera, ou melhor, cenário utilizado para a avaliação do classificador de anomalia. Será necessário garantir que o *autoencoder* aprenda os padrões do conjunto de dados afim de reconstruir a imagem. Caso surja uma imagem fora dessa padrão, *i.e.*

anomalia, o erro da reconstrução será alto por consequência.

Espera-se, então, que as anomalias apresentem um erro alto como característica em comum. Assumindo que as imagens pode pertencer a dois diferentes grupos, o de anomalias e o de imagens normais, o objetivo de cada *autoencoder* é, portanto, construir imagens de erro de anomalias que sejam identificáveis, e distintas das imagens normais, cujas reconstruções apresentarão, idealmente, pouco erro. Além disso, é necessário que seja possível agrupar imagens de erro pertencentes ao mesmo grupo, apesar de possuírem bases diferentes, de acordo com suas características em comum.

Com a garantia de que seja possível agrupar imagens de erro de diferentes bases e identificá-las independentemente de seu contexto original, *i.e* independente de que base cada imagem se origina, o segundo experimento visa, então, encontrar um classificador que faça esse trabalho de forma satisfatória. O classificador deve aprender os padrões das imagens de erro dos dois grupos, e agrupá-las adequadamente.

Primeiro, cada conjunto de dados é dividido entre dois subconjuntos, um somente com não anomalias, chamado de conjunto de treino, e outro contendo ambos os tipos de imagens. O subconjunto com os dois tipos é então dividido entre validação e teste, com cada um dos conjuntos atuando em uma diferente etapa do experimento. É importante ressaltar que nenhum dos conjuntos possui interseção entre si. O número de amostras em cada conjunto deve ser o suficiente para se realizarem os testes de cada rede.

O primeiro experimento será realizado em cada base de dados utilizada. Primeiro é feito o treinamento dos *autoencoders*. Cada base utilizada possui um *autoencoder* específico e cada um deles é representado por uma CNN. Serão feitos treinamentos em vários *autoencoder* diferentes para cada base de dados, visando encontrar aquele de melhor desempenho para cada uma. Os detalhes de como é feita a escolha e o teste de cada será descrito em detalhes mais adiante. A figura 4.1 detalha a metodologia experimental do primeiro experimento.

Após o treino e escolha do melhor *autoencoder*, este deverá operar sobre o seu

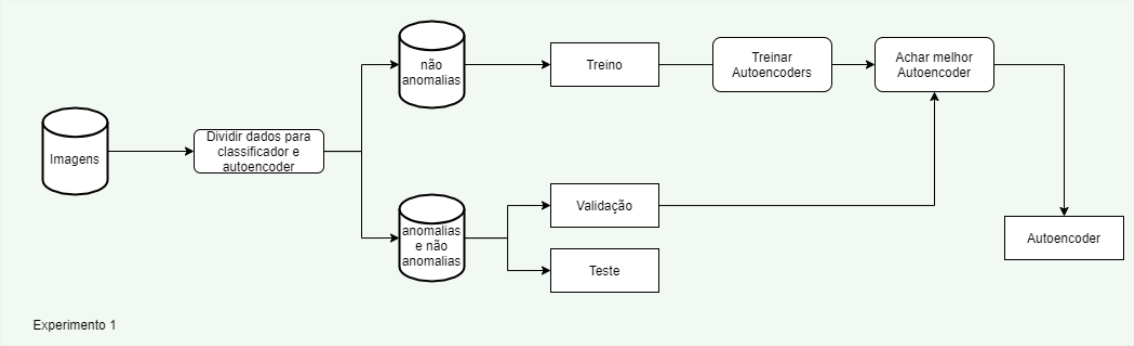

Figura 4.1: A figura detalha a metodologia experimental do experimento I, cujo objetivo é avaliar os *autoencoders* de cada base de dados, a fim de encontrar o mais adequado

respectivo conjunto de testes a fim de reconstruir as imagens. Em seguida, é feito a subtração, pixel por pixel, das imagens originais pelas imagens reconstruídas. As novas imagens de erro geradas serão as imagens de interesse do classificador, e serão imagens utilizadas no segundo experimento.

No segundo experimento, o classificador irá receber todos os conjuntos de imagens de erro, que serão unidos em só um conjunto. Será feita uma análise em diferentes configurações de classificadores, visando obter o que apresentar melhor desempenho baseado nas métricas que serão estabelecidas. A figura 4.2 detalha a metodologia experimental do segundo experimento realizado.

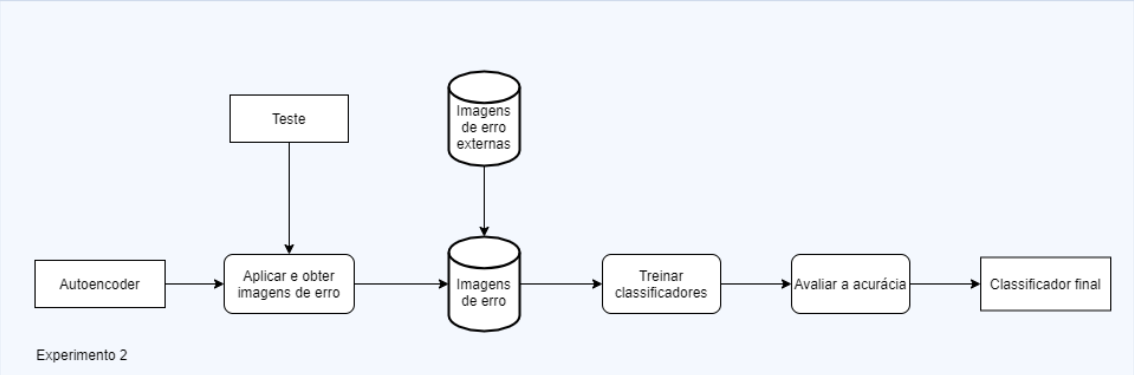

Figura 4.2: A figura representa a metodologia experimental do experimento II, cujo objetivo é analisar o desempenho dos classificadores quando aplicados à base de dados generalizada

4.2 Métricas

A métrica utilizada para análise do desempenho do modelo é a acurácia. A acurácia mede a razão entre a quantidade de acertos do modelo sobre o total de amostras do conjunto de dados. Uma métrica simples, porém eficaz e direta, de avaliar o desempenho de uma rede, principalmente do classificador, por este apresentar um resultado binário, chamado de predição. A predição de uma imagem de erro é, então, 1 se a imagem de erro representar uma anomalia, e 0 caso contrário.

A acurácia é utilizada apenas no resultado do classificador treinado, e não é utilizada em nenhum dos *autoencoders*. Isso ocorre porque o resultado dos *autoencoders*, por ser uma imagem, é muito mais subjetivo, e portanto requer um cuidado especial. Como o resultado final depende apenas do classificador, é assumido que uma boa acurácia no mesmo implica em um bom trabalho dos *autoencoders*.

Outra métrica utilizada foi a função de perda. A função de perda, ou *loss*, representa a distância, ou erro, do resultado final da rede para o resultado desejado. Essa métrica será usada como critério de desempate em casos onde a acurácia de dois ou mais classificadores apresentem o mesmo valor.

4.3 Base de dados

Devido às características do problema, existem poucas bases de dados de qualidade na área. Sultani, Chen e Shah (2018) comentam que a maioria das bases possui ocorrências anômalas que chegam a ser forçadas e não retratam bem a realidade, e que as variações nas anomalias muitas vezes são pequenas. Outro problema que agrava o cenário é de não haver bases específicas para certos problemas específicos, que podem ser interessantes de se analisar.

Mahadevan et al. (2010) disponibilizaram dois dos que provavelmente são os conjunto de dados mais utilizados para detecção de anomalias em vídeo na literatura. As duas bases, peds1 e peds2, contêm *frames* com atividade de pedestres, andando em caminhos diagonais e verticais, respectivamente. Nessas bases, pedestres caminhando

são tratados como comportamento normal, e ciclistas, skatistas e até mesmo carros são tratados como anomalias. As duas já estão previamente organizadas em dois grupos, similarmente à organização utilizada nos experimentos.

A base peds1 possui 70 vídeos, dos quais 32 possuem apenas eventos normais, geralmente utilizados para o treinamento, e 36 vídeos contendo ambos eventos normais e anomalias, utilizados para validação e teste. Convertidos para *frames* individuais, tem-se 6800 imagens para o primeiro grupo de vídeo, sendo todas não anomalias, e 7200 imagens no segundo grupo, das quais 4045 são anomalias, compondo 56% das imagens desse grupo. A figura 4.3 mostra exemplos de imagens da base peds1.

(a) Exemplo de não anomalia da base peds1

(b) Exemplo de anomalia da base peds1

Figura 4.3: Dois exemplos de imagens da bases peds1, com uma imagem normal e uma anomalia, respectivamente. A anomalia em 4.3b é o carro.

A base peds2, um pouco menor, possui 28 vídeos. O primeiro grupo, sem anomalias, possui 16 vídeos e o segundo grupo, 12. Convertidos para imagens, o primeiro grupo possui 2550 imagens e o segundo, 2010, das quais 1648 são anomalias, um total de 81% do grupo. A figura 4.4 mostra exemplos de imagens da base peds2.

Disponibilizada por Sultani, Chen e Shah (2018), a base de dados *UCF-Crime* é uma base de vídeos para detecção de anomalias com diversas situações capturada por câmeras de segurança. As situações incluem roubo, apreensões, incêndios culposos, explosões e outros cenários. Cada situação apresenta vídeos diferentes e para esse trabalho, serão usados vídeos específicos de certas ocorrências. Cada vídeo foi organizado manualmente em dois conjuntos, da mesma forma que as bases peds1 e peds2. O primeiro conjunto possui apenas imagens normais, enquanto que o segundo possui ambos os tipos de imagens.

(a) Exemplo de não anomalia da base peds2

(b) Exemplo de anomalia da base peds2

Figura 4.4: Dois exemplos de imagens da base peds2, com uma imagem normal e uma anomalia, respectivamente. A anomalia em 4.3b são os dois ciclistas e um skatista.

O primeiro vídeo da *UCF-Crime* utilizado foi uma cena de explosão. O vídeo é de uma câmera de segurança capturando imagens de um posto de gasolina. A cena ocorre alguns segundos após um caminhão chegar no posto para ser abastecido. Nesse vídeo, as anomalias compõem o momento desde a explosão do caminhão até o fim do vídeo. A figura 4.5 mostra um exemplo de uma imagem normal e uma anomalia nesse mesmo cenário.

(a) Exemplo de quadro normal retirado do vídeo

(b) Exemplo de quadro anômalo retirado do vídeo

Figura 4.5: Dois exemplos de imagens de uma câmera de segurança em um posto de gasolina. A imagem 4.5a representa uma imagem normal do posto, enquanto que a imagem 4.5b representa um momento após a explosão ter ocorrido.

O segundo vídeo da base utilizado foi um cenário onde uma briga na rua ocorre. O vídeo mostra um cenário comum, com pessoas conversando, e pedestres e carros indo e vindo. Os quadros anômalos compõem uma briga que se inicia subitamente. A figura 4.6 mostra exemplos dos dois tipos de cenários desse vídeo.

O terceiro vídeo utilizado foi um cenário de incêndio culposo. O vídeo mostra imagens de um carro estacionado, até que uma dupla de pessoas chega e ateia fogo dentro do carro. Os exemplos de imagem normal e anômala são mostrados na figura 4.7

(a) Exemplo de quadro normal retirado do vídeo

(b) Exemplo de quadro anômalo retirado do vídeo

Figura 4.6: Duas imagens tiradas de uma câmera de segurança na esquina de uma rua. A imagem 4.6a representa uma imagem normal, com pessoas agindo normalmente. Já a imagem4.6b captura um dos momentos nos quais a briga está ocorrendo.

(a) Exemplo de quadro normal retirado do vídeo

(b) Exemplo de quadro anômalo retirado do vídeo

Figura 4.7: Imagens retiradas de uma câmera de segurança. A imagem 4.7a mostra uma ocorrência normal, com o carro estacionado. A imagem 4.7b mostra uma ocorrência anômala, quando duas pessoas ateiam fogo no carro

4.4 Análise dos Experimentos

Dado os passos descritos na metodologia e o objetivo, os experimentos precisam ser realizados de forma que o modelo possua o melhor desempenho alcançável. Os experimentos serão divididos em duas etapas. A primeira etapa tem o objetivo de

achar o *autoencoder* com o melhor desempenho para cada um dos vídeos nas bases de dados para então, na segunda etapa, usar os *autoencoders* encontrados para construir um classificador geral que obtenha um bom resultado com as imagens geradas.

Os modelos de reconstrução e o classificador possuem diversos hiperparâmetros. Devido a isso, encontrar os parâmetros que resultem no melhor desempenho é crucial, enquanto que a busca manual pela melhor configuração pode se tornar um trabalho extensivo, e de acordo com Liashchynskyi e Liashchynskyi (2019), fazer com que um especialista tenha que tomar bastante decisões quanto ao design da rede, devido ao alto número de possíveis parâmetros e escolhas que é possível encontrar enquanto se constrói uma rede neural. Há, entretanto, uma série de técnicas na literatura que são utilizadas para busca de hiperparâmetros, como o *grid search* e algoritmos genéticos. Para o presente trabalho, foi utilizado o método de *grid search*.

O *grid search* é um método tradicional de busca de hiperparâmetros que simplesmente faz uma busca completa em cima de um dado subconjunto do espaço de hiperparâmetros do algoritmo de treinamento (LIASHCHYNSKYI; LIASHCHYNSKYI, 2019). Desta maneira, dado um número de hiperpâramentros desejados, o *grid search* irá buscar em todas as combinações possíveis, a que apresenta o melhor resultado. A figura 4.8 mostra um espaço de busca do algoritmo *grid search*. O algoritmo é guloso, então é necessário adequar o espaço de busca para que a tarefa não se torne extremamente extensiva e computacionalmente cara.

Os hiperparâmetros e seus respectivos valores utilizados nos experimentos serão baseados no espaço de busca utilizado em (BARROS; NETO, 2020), cujos resultados obtidos comprovam a sua eficácia para experimentos similares. Baseado em seus resultados, será utilizado um subconjunto do espaço proposto em seu trabalho, com alguns hiperparâmetros sendo fixados a algum valor, e outros possuindo um subconjunto menor de possíveis valores, com o objetivo de diminuir o esforço computacional do *grid search*, sem degradar a qualidade dos resultados.

Para todos os experimentos, para fins de classificação, será assumido que todo *frame* anômalo representa um evento individual, *i.e*, que *frames* anômalos não possuam relação entre si, pelo fato dos vídeos nas bases utilizadas possuírem poucos

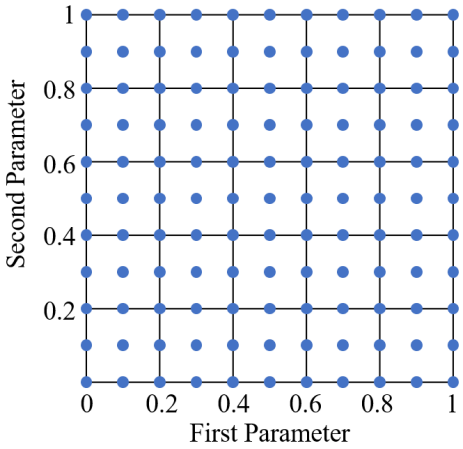

Figura 4.8: Exemplo de um espaço de busca do algoritmo de *grid search* mostrado em Hien, Tien e Hieu (2020). Cada ponto representa uma combinação de hiperparâmetros a ser analisada.

eventos anômalos. Essa abordagem possui limitações que serão discutidas posteriormente, mas uma de suas vantagens é a simplificação do modelo, onde não é preciso saber o contexto dos quadros anteriores e posteriores para se classificar uma imagem.

4.4.1 Experimento I

O objetivo do primeiro experimento é encontrar os melhores hiperparâmetros para cada um dos *autoencoders* encarregados de um conjunto diferente de dados. Foi realizado um *grid search* para diferentes combinações de hiperparâmetros. Será utilizado o conjunto de treino para treinar todos os *autoencoders*, eles então irão realizar a reconstrução de imagens em seus respectivos conjuntos de validação, a fim de serem avaliados por um classificador arbitrário. O *autoencoder* escolhido para cada caso foi aquele cuja reconstrução apresentou maior acurácia no classificador.

O espaço de busca foi organizado como segue na tabela 4.1. A taxa de aprendizado varia entre 0.001 e 0.0001, as funções de ativação variam entre utilizar a função sigmoide ou tangente hiperbólica na saída, e também entre usar $LeakyReLU$ ou $ReLU$ nas camadas intermediárias. As redes também alteram entre a utilização ou não de normalização em lotes, que é uma técnica para treinar redes neurais que normaliza as entradas de uma camada para cada mini lote. A quantidade de filtros da camada intermediária do *autoencoder* foi fixada em 256.

Parâmetros	**Valores**	
Função de ativação	sigmoide	tangente hiperbólica
Taxa de aprendizado	0.001	0.0001
Normalização em lotes	Sim	Não
ReLU	*Leaky*	Normal

Tabela 4.1: Espaço de busca do algoritmo de *grid search* empregado nos *autoencoders*

O experimento foi organizado da seguinte forma: primeiro o *Grid Search* seleciona um conjunto de hiperparâmetros para construir o modelo da rede. Em seguida, é utilizado o conjunto completo de treinamento de cada base de dados para realizar o treinamento de cada *autoencoder*. O *autoencoder*, então, aplica a reconstrução nas imagens dos respectivos conjuntos de validação de cada base, para ser analisada a acurácia do classificador arbitrário quando utilizado com as imagens de erro geradas.

A quantidade de amostras anômalas e não anômalas em cada conjunto de validação foi de 250 cada, somando um total de 500, a priori. Tal número foi escolhido por ser o suficiente para se realizar os experimentos com um número satisfatório no conjunto de validação, e ao mesmo tempo manter um bom número para os conjuntos de testes. As imagens utilizadas no conjunto de validação não serão utilizadas nos próximos experimentos, a fim de não enviesar os resultados.

Os resultados serão expostos a seguir. É importante ressaltar que alguns casos apresentam uma acurácia bastante alta, às vezes até alcançando 100%, isso é devido ao menor número de amostras no teste do classificador. A alta acurácia, porém, não deve afetar negativamente os resultados.

Para a base peds1, a arquitetura escolhida foi a indicada na tabela 4.2. Essa configuração apresentou uma acurácia de 99.8%.

Parâmetros	**Valores**
Função de ativação	sigmoide
Taxa de aprendizado	0.001
Normalização em lotes	Não
ReLU	*Leaky*

Tabela 4.2: Parâmetros do *autoencoder* da base peds1

Para o peds2, foi necessário um tratamento adicional. Foram retiradas 500

não anomalias do primeiro conjunto e adicionadas ao segundo, a fim de balancear um pouco o conjunto de testes, já que a quantidade de não anomalias no segundo grupo era bem pequena. A configuração escolhida foi a mostrada na tabela 4.3, que apresentou uma acurácia de 100%

Parâmetros	**Valores**
Função de ativação	tanh
Taxa de aprendizado	0.001
Normalização em lotes	Sim
ReLU	*Leaky*

Tabela 4.3: Parâmetros do *autoencoder* da base peds2

Para o vídeo de explosão da base *UCF-Crime*, o melhor *autoencoder* gerado pelo *grid search* foi o exposto na tabela 4.4. Nesse caso, a acurácia obtida foi de 100%, e mais de uma configuração apresentou tal resultado. O modelo escolhido, então, foi o que apresentou menor erro entre os resultados.

Parâmetros	**Valores**
Função de ativação	tanh
Taxa de aprendizado	0.001
Normalização em lotes	Sim
ReLU	Normal

Tabela 4.4: Parâmetros do *autoencoder* do primeiro vídeo da base *UCF-Crime*

No vídeo de luta da base *UCF-Crime*, o melhor *autoencoder* foi o mostrado na tabela 4.5. Este apresentou uma acurácia de 100%, e menor erro entre as outras configurações.

Parâmetros	**Valores**
Função de ativação	tanh
Taxa de aprendizado	0.001
Normalização em lotes	Sim
ReLU	*Leaky*

Tabela 4.5: Parâmetros do *autoencoder* do segundo vídeo da base *UCF-Crime*

Por fim, o ultimo vídeo, do cenário de incêndio culposo, tem seu *autoencoder* de melhor desempenho descrito na tabela 4.6. Esta configuração apresentou uma acurácia de 96%.

Parâmetros	Valores
Função de ativação	tanh
Taxa de aprendizado	0.0001
Normalização em lotes	Sim
ReLU	Normal

Tabela 4.6: Parâmetros do *autoencoder* do terceiro vídeo da base *UCF-Crime*

A figura 4.9 mostra a acurácia obtida pelo melhor *autoencoder* de cada um dos conjuntos de dados. UCF-Crime de 1 a 3 se referem aos vídeos de explosão, luta e incêndio, respectivamente. Foi possível observar um alto resultado ao longo de todos conjuntos utilizados, tornando o experimento I satisfatório.

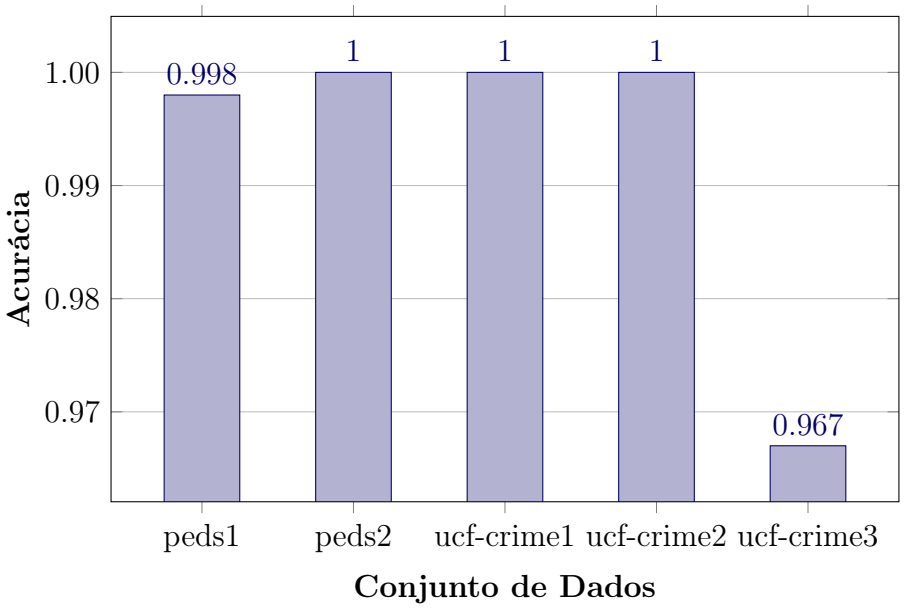

Figura 4.9: Gráfico de barras mostrando a acurácia obtida pelo melhor *autoencoder* encontrado para cada base de dados no experimento I

4.4.2 Experimento II

Com todas as bases de dados possuindo seus respectivos *autoencoders*, será realizado o experimento II. Após o experimento I, serão aplicadas a reconstrução e extração de erro em todos os conjuntos de testes de cada base de dados, e unir todos os conjuntos, formando uma nova base contida apenas de imagens de erro. O objetivo do experimento II é construir um classificador que consiga discriminar o novo conjunto de imagens de erro corretamente, independente do contexto inicial

das imagens.

Para encontrar o classificador com melhor desempenho, será aplicado novamente o algoritmo de *grid search*. A tabela 4.7 mostra como o espaço de busca foi organizado. Os parâmetros taxa de aprendizado, normalização em lotes e *ReLU* são os mesmos utilizados no experimento anterior. O parâmetro taxa de *dropout* indica uma porcentagem dos neurônios a serem desativados em cada camada. A técnica de *dropout* é util para evitar *overfitting*, que é a memorização dos dados de entrada por parte da rede, causando a ilusão de bom desempenho. Por fim, o parâmetro filtros indica a quantidade de filtros em cada uma das três camadas convolucionais do classificador.

Parâmetros	**Valores**		
Taxa de *dropout*	0.1	0.15	0.2
Taxa de aprendizado	0.001	0.0001	
Normalização em lotes	Sim	Não	
ReLU	*Leaky*	Normal	
Filtros	16,32,64	32,64,128	

Tabela 4.7: Espaço de busca do algoritmo de *grid search* empregado no classificador do experimento II

Após a execução do *grid search*, o melhor resultado obtido foi de 98.4% de acurácia, pela configuração mostrada na tabela 4.8.

Parâmetros	**Valores**
Taxa de *dropout*	0.15
Taxa de aprendizado	0.0001
Normalização em lotes	Sim
ReLU	Normal
Filtros	16,32,64

Tabela 4.8: Parâmetros do classificador construído no experimento II

Analisando o desempenho do classificador sobre cada um dos *datasets* individualmente, foram obtidos os resultados mostrados na figura 4.10. UCF-Crime de 1 a 3 representam os vídeos de explosão, luta e incêndio, respectivamente. É possível notar que os resultados foram bem parecidos com os obtidos no experimento I, com uma acurácia alta em todas as bases, sendo o vídeo de explosão o que apresentou maior discrepância entre as acurácias.

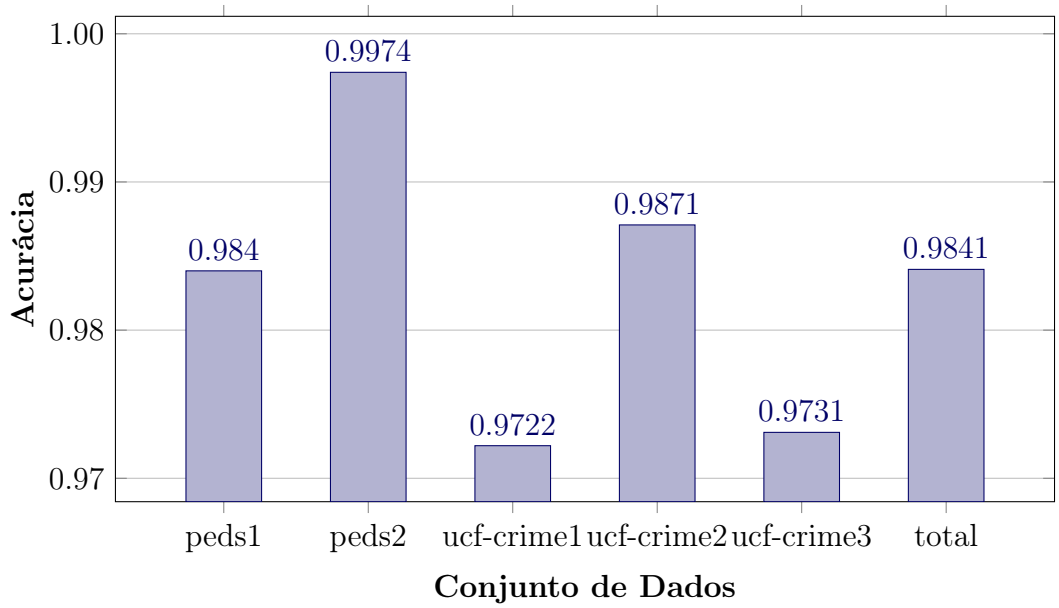

Figura 4.10: Gráfico de barras mostrando a acurácia do classificador em cada base de dados individualmente

Ainda com as análises individuais em cada conjunto de dados, a tabela 4.9 mostra a quantidade de imagens classificadas de forma errada, sobre o total de imagens usadas para o teste do classificador em cada conjunto. A última linha mostra a análise dos erros e total de imagens sobre o conjunto unificado.

Conjuntos	Erros	Total de Imagens	Acurácia
peds1	22	1376	0.9840
peds2	1	392	0.9974
ucf-crime1	9	324	0.9722
ucf-crime2	2	156	0.9871
ucf-crime3	4	149	0.9731
conjunto total	38	2397	0.9841

Tabela 4.9: Quantidade de erros individuais e erros consecutivos por conjunto de dados do classificador do experimento II

Uma ocorrência onde há má classificação é a de um falso positivo, onde uma imagem normal é classificada como anômala, em quadros que antecedem diretamente eventos anômalos. A figura 4.11 apresenta a única imagem do conjunto peds2 que foi mal classificada. A imagem é, pelo agrupamento original do conjunto, uma não anomalia, porém é possível observar uma bicicleta no canto esquerdo da imagem, indicando o início de um evento anômalo.

Como o modelo aprende que a presença de bicicletas implica em anomalia, o quadro acaba tendo uma classificação ambígua, possivelmente oriunda da própria construção do conjunto. Logo, o erro é, de certa forma, justificável. Espera-se que tal erro seja prevenido se forem eliminadas as ambiguidades na construção dos conjuntos de dados utilizados.

Figura 4.11: A figura apresenta a imagem do conjunto de dados peds2 que foi classificada de forma errada pelo classificador

Capítulo 5

Conclusão

Nesse capítulo serão apresentadas as conclusões obtidas dos experimentos, baseadas na proposta inicial do trabalho. Aqui também serão discutidas algumas limitações encontradas no trabalho, bem como possíveis trabalhos futuros que podem eliminar tais limitações.

5.1 Considerações finais

A proposta do trabalho foi a de construir um modelo de detecção de anomalia em vídeos que conseguisse discriminar seus quadros individualmente da forma mais precisa possível, sem conhecer o contexto original do mesmo. Foi utilizado um modelo contendo *autoencoders* convolucionais para reconstrução de imagens e um classificador que tem como objetivo agrupá-las corretamente em anomalias ou não anomalias.

No geral, ao longo do trabalho foi possível perceber que o modelo proposto, de *autoencoder* seguido por classificador, é de fato preciso. No primeiro experimento, além do melhor *autoencoder* escolhido, outros *autoencoder* no mesmo espaço de busca apresentaram resultados similares. Por exemplo, nas bases peds1 e peds2, nenhum dos *autoencoders* utilizados no *grid search* apresentou acurácia menor que 90%.

Os resultados obtidos no segundo experimento também foram bastante satisfató-

rios. Foi possível obter uma alta acurácia no modelo, mesmo com a união de diversos conjuntos de dados. Isso mostra que o modelo é capaz de discriminar as imagens de erro corretamente, apenas tendo conhecimento de outras imagens similares, e não de seu contexto original.

5.2 Limitações e trabalhos futuros

Apesar de resultados satisfatórios, ao longo do trabalho foram encontradas algumas dificuldades relacionadas a certas limitações do modelo. Além disso, ainda há muito espaço de melhorias para o mesmo, a fim de aumentar não só a acurácia, como também a confiabilidade do mesmo. Algumas limitações e possíveis formas de prevenções futuras serão discutidas a seguir.

Um dos possíveis trabalhos futuros testar a acurácia do modelo já treinado ao receber novos dados nunca antes vistos, de uma base totalmente diferente, a fim de observar a sua adaptabilidade. É esperado, então, que estudos futuros busquem construir um modelo que apresente uma classificação precisa mesmo em cenários inéditos, possibilitando uma maior democratização do mesmo.

Uma das limitações enfrentadas foi a dificuldade em encontrar bases de dados para a realização do trabalho. Embora essa dificuldade tenha sido tratada de certa forma com a criação manual de bases a partir de vídeos da UCF-Crime, isso resultou em uma possível dificuldade quanto a diferença estrutural das imagens deste se comparadas às imagens das bases peds1 e peds2. Uma abordagem futura poderia ser, então, um modelo que padronize as estruturas de tais imagens na entrada do mesmo, a fim de facilitar a generalização.

Outra limitação foi o tratamento de quadros de forma individual. Como mencionado antes, tratar quadros do vídeo de forma que os mesmo não tenham relação entre si foi uma forma de simplificar o modelo. Essa suposição faz com que quadros não possuam informações adicionais que podem ser úteis para aumentar a precisão do modelo, como sobre o evento anômalo no qual estão inseridos, ou a com quais quadros estão relacionados.

Uma proposta para o futuro é a adição do elemento temporal nos quadros. Ter as informações sobre os eventos anômalos em sua totalidade, e conhecimento de quadros relacionados entre si em um fluxo temporal pode ser uma ferramenta útil na tentativa de aumentar a acurácia do modelo. Dessa forma, a classificação de um quadro poderia ser positivamente influenciada pelo fluxo do tempo em um vídeo, já que um quadro logo antes e depois de anomalias tem alta probabilidade de também ser uma anomalia.

Referências

AGGARWAL, C. C. *An Introduction to Outlier Analysis*. Springer International Publishing, 2017. 1-34 p. Disponível em: <http://link.springer.com/10.1007/978-3-319-47578-3_1>.

ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. In: . [S.l.: s.n.], 2018. v. 2018-January.

BARBIERI, J. et al. Simulating real profiles for shilling attacks: A generative approach. *Knowledge-Based Systems*, v. 230, 2021. ISSN 09507051.

BARROS, M. O. de; NETO, N. A. L. *Classificação Automática de Anomalia em Vídeo*. 2020.

CHALAPATHY, R.; CHAWLA, S. *Deep Learning for Anomaly Detection: A Survey*. 2019.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. *Anomaly detection: A survey*. 2009.

COLLETT, D.; LEWIS, T. The subjective nature of outlier rejection procedures. *Applied Statistics*, v. 25, 1976. ISSN 00359254.

FRéNAY, B.; VERLEYSEN, M. Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, v. 25, 2014. ISSN 21622388.

GARDNER, M. W.; DORLING, S. R. Artificial neural networks (the multilayer perceptron) - a review of applications in the atmospheric sciences. *Atmospheric Environment*, v. 32, 1998. ISSN 13522310.

GOLDBERG, X. Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, v. 6, 2009. ISSN 19394608.

HIEN, N. L. H.; TIEN, T. Q.; HIEU, N. V. Web crawler: Design and implementation for extracting article-like contents. *Cybernetics and Physics*, v. 9, 2020. ISSN 22264116.

HODGE, V. J.; AUSTIN, J. *A survey of outlier detection methodologies*. 2004.

HOMAYOUNI, H. et al. Anomaly detection in covid-19 time-series data. *SN Computer Science*, v. 2, 2021. ISSN 2662-995X.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, v. 60, 2017. ISSN 15577317.

LIASHCHYNSKYI, P.; LIASHCHYNSKYI, P. *Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS*. 2019.

MAHADEVAN, V. et al. Anomaly detection in crowded scenes. In: . [S.l.]: IEEE, 2010. ISBN 978-1-4244-6984-0.

MILJKOVIć, D. Review of novelty detection methods. In: . [S.l.: s.n.], 2010.

MINSKY, M.; PAPERT, S. Perceptron: an introduction to computational geometry. *The MIT Press, Cambridge, expanded edition*, v. 19, 1969.

MITCHELL, T. M. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, v. 45, 1997.

MOHAMED, H. et al. Assessment of artificial neural network for bathymetry estimation using high resolution satellite imagery in shallow lakes : Case study el burullus lake . *International Water Technology Conference*, 2015.

NAQA, I. E.; MURPHY, M. J. *What Is Machine Learning?* 2015.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, 1958. ISSN 0033295X.

RUDER, S. An overview optimization gradients. *arXiv preprint arXiv:1609.04747*, 2017. ISSN 0006341X.

SATHYA, R.; ABRAHAM, A. Comparison of supervised and unsupervised learning algorithms for pattern classification. *International Journal of Advanced Research in Artificial Intelligence*, v. 2, 2013. ISSN 21654050.

SCHLEGL, T. et al. f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical Image Analysis*, v. 54, 2019. ISSN 13618423.

SHARMA, S.; SHARMA, S.; ATHAIYA, A. Activation functions in neural networks. *International Journal of Engineering Applied Sciences and Technology*, v. 04, 2020. ISSN 2455-2143.

SULTANI, W.; CHEN, C.; SHAH, M. Real-world anomaly detection in surveillance videos. In: . [S.l.: s.n.], 2018. ISSN 10636919.

WERBOS, P. J. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. Tese (Doutorado) — Harvard University, 1974.

WONG, W. K. et al. Rule-based anomaly pattern detection for detecting disease outbreaks. In: . [S.l.: s.n.], 2002.

XU, D. et al. Learning deep representations of appearance and motion for anomalous event detection. In: . [S.l.: s.n.], 2015.

XU, D. et al. Detecting anomalous events in videos by learning deep representations of appearance and motion. *Computer Vision and Image Understanding*, v. 156, 2017. ISSN 1090235X.

YAO, D. D. et al. Anomaly detection as a service: Challenges, advances, and opportunities. *Synthesis Lectures on Information Security, Privacy, and Trust*, v. 9, 2017. ISSN 1945-9742.

YAO, X.; LIU, Y. Machine learning. In: *Search Methodologies*. [S.l.]: Springer, 2014. p. 477–517.

YOU, W. et al. An intelligent deep feature learning method with improved activation functions for machine fault diagnosis. *IEEE Access*, v. 8, 2020. ISSN 21693536.

ZHAI, J. et al. Autoencoder and its various variants. In: . [S.l.: s.n.], 2019.

ZHOU, C.; PAFFENROTH, R. C. Anomaly detection with robust deep autoencoders. In: . [S.l.: s.n.], 2017. Part F129685.

ZONG, B. et al. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: . [S.l.: s.n.], 2018.