Pontifícia Universidade Católica
do Rio de Janeiro

Amanda Lucas Pereira

Aprendizado Semi e Auto-supervisionado aplicado à classificação multi-label de imagens de inspeções submarinas

Dissertação de Mestrado

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós–graduação em Engenharia Elétrica, do Departamento de Engenharia Elétrica da PUC-Rio.

Orientador : Prof. Marco Aurélio Cavalcanti Pacheco
Coorientador: Dra. Manoela Rabello Kohler

Rio de Janeiro
Março de 2023

PUC-Rio - Certificação Digital Nº 2112316/CA

Pontifícia Universidade Católica
do Rio de Janeiro

Amanda Lucas Pereira

Aprendizado Semi e Auto-supervisionado aplicado à classificação multi-label de imagens de inspeções submarinas

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós–graduação em Engenharia Elétrica da PUC-Rio. Aprovada pela Comissão Examinadora abaixo:

Prof. Marco Aurélio Cavalcanti Pacheco
Orientador
Departamento de Engenharia Elétrica – PUC-Rio

Dra. Manoela Rabello Kohler
Coorientador
Departamento de Engenharia Elétrica – PUC-Rio

Prof. Leonardo Alfredo Forero Mendoza
UERJ

Dra. Ana Carolina Alves Abreu
Departamento de Engenharia Elétrica – PUC-Rio

Rio de Janeiro, 10 de Março de 2023

PUC-Rio - Certificação Digital Nº 2112316/CA

Todos os direitos reservados. A reprodução, total ou parcial do trabalho, é proibida sem a autorização da universidade, do autor e do orientador.

Amanda Lucas Pereira

Graduou-se em Engenharia de Controle e Automação pelo Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo no ano de 2020.

PUC-Rio - Certificação Digital Nº 2112316/CA

Ficha Catalográfica

Lucas Pereira, Amanda

Aprendizado Semi e Auto-supervisionado aplicado à classificação multi-label de imagens de inspeções submarinas / Amanda Lucas Pereira; orientador: Marco Aurélio Cavalcanti Pacheco; coorientador: Manoela Rabello Kohler. – 2023.

68 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica, 2023.

Inclui bibliografia

1. Engenharia Elétrica – Teses. 2. inspeções submarinas. 3. classificação de imagem. 4. classificação multi-label. 5. aprendizado auto-supervisionado. 6. aprendizado semi-supervisionado. I. Cavalcanti Pacheco, Marco Aurélio. II. Rabello Kohler, Manoela. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. IV. Título.

CDD: 621.3

PUC-Rio - Certificação Digital Nº 2112316/CA

Dedicado à minha mãe,
pelo seu imensurável suporte.

Agradecimentos

Primeiramente, gostaria de agradecer ao meu orientador e à minha co-orientadora pelo apoio durante todo o curso e principalmente durante a execução da dissertação.

Agradeço minha mãe, Mercedes, pelo suporte que têm me dado ao longo da minha vida. Também agradeço ao meu irmão Burgie e minha irmã Aline. Sem o apoio da minha família não sei se conseguiria ter trilhado o caminho que me trouxe até aqui.

Gostaria de agradecer também aos meus colegas e amigos do ICA, pelo incentivo e pela troca de ideias e conhecimento.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001. Adicionalmente, o projeto também recebeu apoio financeiro da Petrobras, Petróleo Brasileiro S.A.

Os experimentos realizados foram possíveis graças às máquinas disponibilizadas pelo Laboratório Nacional de Computação Científica - LNCC.

PUC-Rio - Certificação Digital Nº 2112316/CA

Resumo

Lucas Pereira, Amanda; Cavalcanti Pacheco, Marco Aurélio; Rabello Kohler, Manoela. **Aprendizado Semi e Auto-supervisionado aplicado à classificação multi-label de imagens de inspeções submarinas**. Rio de Janeiro, 2023. 68p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

O segmento *offshore* de produção de petróleo é o principal produtor nacional desse insumo. Nesse contexto, inspeções submarinas são cruciais para a manutenção preventiva dos equipamentos, que permanecem toda a vida útil em ambiente oceânico. A partir dos dados de imagem e sensor coletados nessas inspeções, especialistas são capazes de prevenir e reparar eventuais danos. Tal processo é profundamente complexo, demorado e custoso, já que profissionais especializados têm que assistir a horas de vídeos atentos a detalhes. Neste cenário, o presente trabalho explora o uso de modelos de classificação de imagens projetados para auxiliar os especialistas a encontrarem o(s) evento(s) de interesse nos vídeos de inspeções submarinas. Esses modelos podem ser embarcados no ROV ou na plataforma para realizar inferência em tempo real, o que pode acelerar o ROV, diminuindo o tempo de inspeção e gerando uma grande redução nos custos de inspeção. No entanto, existem alguns desafios inerentes ao problema de classificação de imagens de inspeção submarina, tais como: dados rotulados balanceados são caros e escassos; presença de ruído entre os dados; alta variância intraclasse; e características físicas da água que geram certas especificidades nas imagens capturadas. Portanto, modelos supervisionados tradicionais podem não ser capazes de cumprir a tarefa. Motivado por esses desafios, busca-se solucionar o problema de classificação de imagens submarinas a partir da utilização de modelos que requerem menos supervisão durante o seu treinamento. Neste trabalho, são explorados os métodos DINO (*Self-DIstillation with NO labels*, auto-supervisionado) e uma nova versão multi-label proposta para o PAWS (*Predicting View Assignments With Support Samples*, semi-supervisionado), que chamamos de mPAWS (multi-label PAWS). Os modelos são avaliados com base em sua performance como extratores de *features* para o treinamento de um classificador simples, formado por uma camada densa. Nos experimentos realizados, para uma mesma arquitetura, se obteve uma performance que supera em 2.7% o f1-score do equivalente supervisionado.

Palavras-chave

inspeções submarinas; classificação de imagem; classificação multi-label; aprendizado auto-supervisionado; aprendizado semi-supervisionado.

PUC-Rio - Certificação Digital Nº 2112316/CA

Abstract

Lucas Pereira, Amanda; Cavalcanti Pacheco, Marco Aurélio (Advisor); Rabello Kohler, Manoela (Co-Advisor). **Semi and Self-supervised Learning applied to the multi-label classification of underwater inspection images**. Rio de Janeiro, 2023. 68p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

The *offshore* oil production segment is the main national producer of this input. In this context, underwater inspections are crucial for the preventive maintenance of equipment, which remains in the ocean environment for its entire useful life. From the image and sensor data collected in these inspections, experts are able to prevent and repair damage. Such a process is deeply complex, time-consuming and costly, as specialized professionals have to watch hours of videos attentive to details. In this scenario, the present work explores the use of image classification models designed to help experts to find the event(s) of interest in underwater inspection videos. These models can be embedded in the ROV or on the platform to perform real-time inference, which can speed up the ROV, monitor notification time, and greatly reduce verification costs. However, there are some challenges inherent to the problem of classification of images of armored submarines, such as: balanced labeled data are expensive and scarce; the presence of noise among the data; high intraclass variance; and some physical characteristics of the water that achieved certain specificities in the captured images. Therefore, traditional supervised models may not be able to fulfill the task. Motivated by these challenges, we seek to solve the underwater image classification problem using models that require less supervision during their training. In this work, they are explorers of the DINO methods (Self-Distillation with NO labels, self-supervised) and a new multi-label version proposed for PAWS (Predicting View Assignments With Support Samples, semi-supervised), which we propose as mPAWS (multi-label PAWS). The models are evaluated based on their performance as *features* extractors for training a simple classifier, formed by a dense layer. In the experiments carried out, for the same architecture, a performance was obtained that exceeds by 2.7% the f1-score of the supervised equivalent.

Keywords

underwater inspections; image classification; multi-label classification; self-supervised learning; semi-supervised learning.

PUC-Rio - Certificação Digital Nº 2112316/CA

Sumário

PUC-Rio - Certificação Digital Nº 2112316/CA

PUC-Rio - Certificação Digital Nº 2112316/CA

Lista de figuras

PUC-Rio - Certificação Digital Nº 2112316/CA

Lista de tabelas

PUC-Rio - Certificação Digital Nº 2112316/CA

Lista de Abreviaturas

AUV – *Autonomous Underwater Vehicle*

CV – *Computer Vision*

DINO – *Self-DIstillation with NO labels*

FN – *False Negative*

FP – *False Positive*

ML – *Machine Learning*

mPAWS – multi-label PAWS

PAWS – *Predicting View Assignments With Support Samples*

ROV – *Remoted Operated Vehicle*

SOTA – *State-of-the-art*

TN – *True Negative*

TP – *True Positive*

ViT – *Vision Transformer*

PUC-Rio - Certificação Digital Nº 2112316/CA

PUC-Rio - Certificação Digital Nº 2112316/CA

A perplexidade é o início do conhecimento.

Khalil Gibran

1
Introdução

1.1
Motivação

A utilização de equipamentos *offshore* pela indústria de petróleo e gás gera a necessidade de uma avaliação periódica de suas condições. Nesse contexto, *offshore*, se refere à produção realizadana em costa marítima, enquanto *onshore* são conhecidos por estarem localizados em terra. Durante sua vida útil, esses equipamentos permanecem em águas profundas, operando no complexo e desafiador ambiente marinho [1]. Além da possibilidade de ocorrência de eventos capazes de danificar diretamente essas máquinas, algumas propriedades da água como salinidade e pH também podem ser igualmente agressivas aos seus componentes [2]. Equipamentos defeituosos apresentam maior risco de desastres ambientalmente e materialmente prejudiciais, podendo também envolver incidentes com humanos [3]. Além disso, as empresas buscam continuamente obter um maior lucro por meio do aumento da produtividade e da vida útil dos equipamentos. Portanto, é fundamental que a manutenção preventiva seja realizada com diligência, a fim de manter todos os maquinários em condições adequadas de trabalho.

Devido às condições ambientais desafiadoras encontradas onde tais equipamentos são instalados, a manutenção preventiva é feita através de monitoramento remoto por vídeo, um processo chamado *inspeção submarina* [1]. Nessas inspeções, dados de imagens e sensores são coletados por veículos submarinos operados remotamente (*Remotely Operated Vehicle*, ROV) ou veículos submarinos autônomos (*Autonomous Underwater Vehicle*, AUV) que viajam até o fundo do mar [4]. Os dados da imagem são compostos por vídeos de dutos, acessórios de dutos, plataformas, estruturas estáticas e outros tipos de equipamentos submarinos. Após a filmagem, os dados coletados são analisados visualmente por profissionais altamente especializados a fim de avaliar a integridade desses equipamentos, relatando qualquer dano externo que possa levar a uma futura falha durante a operação. Esse processo é longo e elaborado, exigindo alta concentração e expertise dos técnicos para inspecionar uma coleção imensa de vídeos. Uma solução para essa trabalhosa tarefa seria automatizar

PUC-Rio - Certificação Digital Nº 2112316/CA

parte do processo, o que pode ser feito através da implementação de métodos de classificação de imagens para identificar objetos e eventos em cada *frame* dos vídeos.

A utilização de um modelo para classificação automática dos objetos pode trazer grandes benefícios para as empresas do setor de óleo e gás. Tais modelos podem ser facilmente embarcados no ROV ou na plataforma e, ao realizarem inferências em tempo real, podem auxiliar na operação remota do ROV. Essa melhoria pode aumentar a eficiência das inspeções submarinas, pois tem o potencial de reduzir o tempo de filmagem da inspeção. Além disso, uma redução no tempo de inspeção se traduz diretamente numa contração nos custos de inspeção, o que é algo positivo para grandes empresas que possuem uma quantidade considerável de equipamentos *offshore* que precisam de manutenção.

Há uma variedade de problemas de classificação de imagens resolvidos por modelos de aprendizado profundo [5]. Tais aplicações geralmente envolvem métodos de aprendizado supervisionado, nos quais o treinamento é feito expondo os modelos a dados rotulados [6]. No entanto, alguns fatores podem limitar o desempenho de tais modelos, como a disponibilidade de dados rotulados e balanceados, a complexidade das classes subjacentes e o ruído. Portanto, existem alguns grandes desafios técnicos ao aplicar esses modelos para classificar as imagens de inspeção submarina.

O primeiro é o alto custo e ruído associados ao processo de rotulagem de imagens de inspeção submarina. Os modelos supervisionados dependem muito da qualidade e disponibilidade de dados rotulados balanceados [7]. Consequentemente, os modelos de classificação de imagens supervisionados de última geração, considerados o atual *state-of-the-art* (SOTA), comprometem-se com a disponibilidade de milhares de dados rotulados balanceados disponíveis publicamente para um desempenho satisfatório. Existem dois problemas principais para nossa aplicação: escassez e ruído dos rótulos.

Imagens de inspeção rotuladas são caras e escassas, pois demandam pessoal especializado que rotula um pequeno número de *frame*s de cada vídeo. Além disso, algumas classes podem ser ruidosas enquanto outras são bem definidas. A rotulagem é feita por um grupo de pessoas qualificadas e treinadas, mas com formações e experiências diferentes. Por exemplo, alguns funcionários podem não estar familiarizados com todos os equipamentos que formam uma tubulação de extração submarina [8]. Além disso, o processo é cansativo, o que pode levar ao erro humano de perder um evento em um *frame* e, portanto, adicionar ruído.

Em segundo lugar, existem desafios inerentes à classificação de imagens

PUC-Rio - Certificação Digital Nº 2112316/CA

submarinas. Algumas propriedades decorrentes do fato do meio de propagação da luz ser a água do mar afetam diretamente a qualidade da imagem, como: difração e absorção da luz, e a atenuação sofrida por diferentes comprimentos de onda [9]. Apesar da distorção mais complexa como resultado de espalhamento e refração, as condições de iluminação mudam substancialmente de acordo com a profundidade onde cada imagem é coletada [10]. Adicionalmente, o oceano profundo apresenta *neve marinha* que é uma chuva constante de sedimentos principalmente orgânicos [11] que podem refletir as luzes artificiais instaladas nos ROV/AUVs, colaborando para a produção de uma textura irregular na imagem final [12].

Finalmente, os dados de imagem de inspeção submarina são mais complexos do que os conjuntos de pesquisa clássicos. Os modelos SOTA para classificação de imagem são comparados principalmente em conjuntos de dados famosos como ImageNet [13] e CIFAR10 [14], onde cada amostra é uma imagem de rótulo único, alinhada e centralizada no ponto de vista de sua captura. Além disso, as imagens apresentam resolução satisfatória, a oclusão do objeto principal é quase inexistente e as classes são bem balanceadas. Portanto, existe uma certa padronização nesses conjuntos. O mesmo não pode ser assumido quando se trata de imagens de inspeções submarinas.

Existem alguns desafios em relação aos próprios dados de imagem de inspeção. A primeira vem do fato de que as imagens de inspeção não são fotografias de captura única como nos *benchmarks* citados acima, mas *frames* amostrados de vídeos. Isso significa que pode-se esperar que um objeto, como um acessório de tubulação, seja apresentado em várias visualizações em diferentes *frames*, e cada visualização pode apresentar diferentes condições, por exemplo, resolução, iluminação, etc. Todas as visualizações do mesmo objeto devem ser classificadas corretamente. Além disso, alguns *frames* podem ficar desfocados devido ao movimento do veículo e como faltam *frames* marcados, estes não serão descartados.

Há também uma alta variabilidade intraclasse e um alto desbalanceamento de classe nos dados. Nesse contexto, variabilidade intraclasse se refere à diversidade de objetos, ou de características dos objetos, pertencentes a uma mesma classe. Uma das justificativas para essa característica é que os equipamentos *offshore* podem vir de diversos fornecedores, sendo que cada um pode oferecer um determinado objeto em formas, cores e tamanhos diferentes. Há também uma variabilidade adicionada pela data em que os vídeos são gravados, pois alguns padrões do setor podem mudar com o tempo. Existe também uma variação no contexto, pois objetos da mesma classe podem desempenhar funções diferentes e, consequentemente, apresentar formas variadas [15]. Além

PUC-Rio - Certificação Digital Nº 2112316/CA

disso, o desbalanceamento de classe corresponde a um desafio inerente à aplicação. A maioria dos *frames* de vídeo mostra objetos em boas condições, havendo diferença considerável no número de ocorrências para classes importantes. Por exemplo, *frames* que retratam dutos que conectam diferentes plataformas são mais frequentes do que acessórios usados em conexões de encaixe.

Os desafios técnicos apresentados acima podem limitar o desempenho de modelos supervisionados para classificação de imagens de inspeção submarina. Além disso, embora as empresas tenham milhares de vídeos de inspeção, apenas uma parte mínima dos *frames* é rotulada e há uma grande disponibilidade de dados não rotulados para uso. Nesse contexto, explorar modelos menos dependentes de rótulos pode ter um efeito positivo.

O aprendizado semi-supervisionado compreende métodos em que os modelos são capazes de extrair conhecimento de dados rotulados e não rotulados. Em geral, a fração rotulada dos dados é consideravelmente menor do que a não rotulada. O método semissupervisionado conhecido como PAWS [16] propõe uma etapa de pré-treinamento com perda contrastiva entre imagens marcadas e não marcadas, o que permite que os modelos alcancem resultados SOTA. Por outro lado, o aprendizado autossupervisionado não requer nenhum rótulo para treinar o modelo. Tais métodos fazem uso de uma busca automática por sinais supervisionados nos dados fornecidos. O método auto supervisionado conhecido como DINO [17] propõe usar a destilação do conhecimento no pré-treinamento, alcançando $80,1\%$ top-1 com Vision Transformers (ViT) pré-treinados [18]. Um ganho de desempenho semelhante também foi observado em configurações semi-supervisionadas, especialmente no que os autores chamam de eficiência do rótulo [19], [20]. Em problemas como o proposto, onde há abundância de imagens não rotuladas disponíveis para serem usadas para treinamento, a aplicação de semi e auto-supervisão pode levar a resultados interessantes.

Motivado pelos argumentos acima, este trabalho aborda o problema de classificação multi-rótulo de imagens de inspeção submarina. Os frameworks testados foram o DINO e uma nova versão multi-label do PAWS, que chamamos de mPAWS (multi-label PAWS). Como o problema em questão trata de dados multi-rótulo – o que os originais não –, foram implementadas modificações para se adequar ao nosso pipeline, propondo um novo método que permite sua aplicação direta em cenários multi-rótulo.

1.2 Objetivos

PUC-Rio - Certificação Digital Nº 2112316/CA

1.2.1
Objetivo Geral

O objetivo principal deste trabalho é solucionar o problema de classificação multi-rótulo de imagens de inspeção submarina.

1.2.2
Objetivos Específicos

Os objetivos específicos do presente trabalho estão listados a seguir.

1. Realizar uma revisão bibliográfica de modelos SOTA para classificação de imagens de inspeções submarinas.

2. Propor um novo método mPAWS para classificação de imagens *multi-label*, baseado no método multi-classe PAWS.

3. Avaliar de forma quantitativa e qualitativa os métodos DINO e mPAWS, assim como métodos *baseline* supervisionados, para a resolução do problema de classificação de imagens de inspeções submarinas.

1.3
Contribuições

As principais contribuições deste trabalho estão listadas a seguir.

1. Desenvolvimento de um novo método mPAWS para classificação de imagens *multi-label*.

2. Realização de uma avaliação quantitativa dos métodos testados através de experimentos com um classificador simples.

3. Realização de uma avaliação qualitativa dos métodos testados através de experimentos com um conjunto de dados construído a partir de imagens públicas de inspeções submarinas.

1.4
Organização do Documento

O presente trabalho está organizado da seguinte forma. Primeiramente, é realizada uma introdução ao procedural de inspeções submarinas no capítulo 2. Na sequência, uma revisão da literatura apresenta conceitos básicos e trabalhos relacionados no capítulo 3. Em seguida, no capítulo 4, são apresentados os materiais e métodos. No capítulo 5 são apresentados e discutidos os experimentos realizados e os resultados obtidos. Finalmente, no capítulo 6, as conclusões e trabalhos futuros são delineados.

PUC-Rio - Certificação Digital Nº 2112316/CA

2
Inspeções Submarinas

A maior parte do petróleo produzido no Brasil é produto de instalações *offshore*. No ano de 2022, a produção *offshore* correspondeu a cerca de 92% do total de barris de óleo equivalente por (boe) (Tabela 2.1). Logo, é indubitável a relevância econômica e energética da produção em alto mar quando comparada à produção em terra. Porém, mesmo que a produção *offshore* seja um processo mais produtivo, este apresenta complexidades que elevam seu custo no que diz respeito a tecnologia utilizada e a logística dos insumos extraídos.

Tabela 2.1: Produção Comparada de Petróleo no Brasil no ano de 2022 [21].

Ambiente	Petróleo Equivalente (boe)
Produção Pré-sal	5.691.034.854,56
Produção Pós-sal Mar	3.834.807.355,34
Produção Terra	776.719.104,39
Total	10.302.561.314,30

PUC-Rio - Certificação Digital Nº 2112316/CA

Um sistema de produção *offshore* compreende diversos equipamentos, responsáveis pela perfuração, extração, escoamento e armazenamento do óleo extraído. Se trata de uma área multidisciplinar, e existe uma variedade de soluções possíveis para que seja possível realizar a produção em alto mar. Um esquemático de uma configuração está apresentado na Figura 2.1. Cada estrutura deve estar em perfeito estado de funcionamento para que a produção ocorra de maneira segura e eficiente.

O processo de acompanhamento do estado de funcionamento e vida útil de equipamentos *offshore* é realizado através de um processo chamado *inspeção submarina*. Essas inspeções também são utilizadas em outras aplicações da indústria de petróleo, como por exemplo no processo de certificação de plataformas na qual verifica-se se toda a planta do projeto está de acordo com normas nacionais e internacionais [22]. O presente trabalho realiza a classificação de imagens de inspeções submarinas realizadas para manutenção de equipamentos e estruturas submarinas – sejam essas plataformas, dutos, acessórios, etc –, logo esse será o foco do texto.

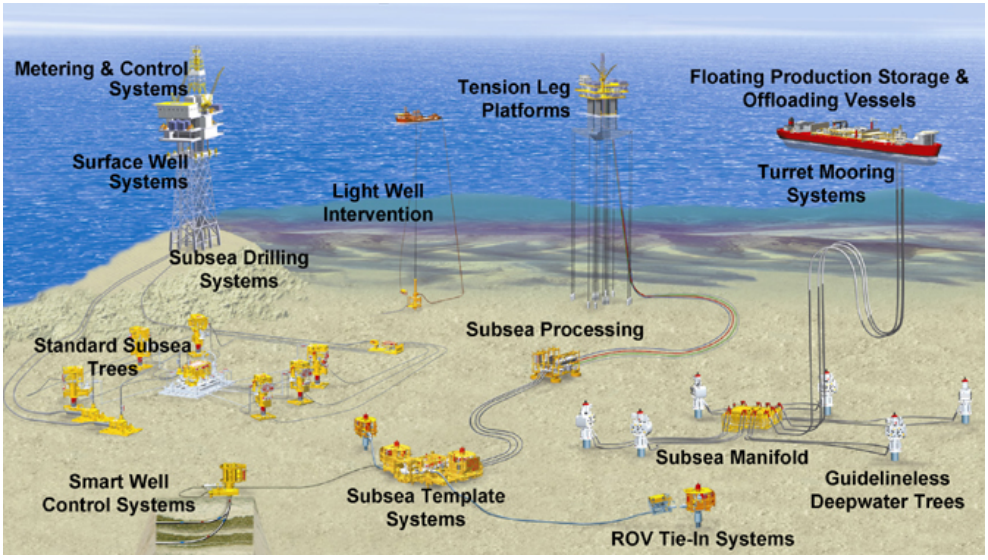

Figura 2.1: Esquemtático dos equipamentos presentes em um sistema de produção *offshore*. [23]

O processo de inspeção submarina pode ser realizado por mergulhadores ou veículos submarinos. A primeira abordagem era a única utilizada durante muito tempo, porém, com o início da extração de óleo e gás em águas profundas, tornou-se inviável manter somente humanos nesse processo. Atualmente, mergulhadores realizam atividades de inspeção apenas em situações de baixa profundidade, enquanto que AUVs e ROVs se estabeleceram como a solução padrão para inspecionamento em águas profundas (Figura 2.2).

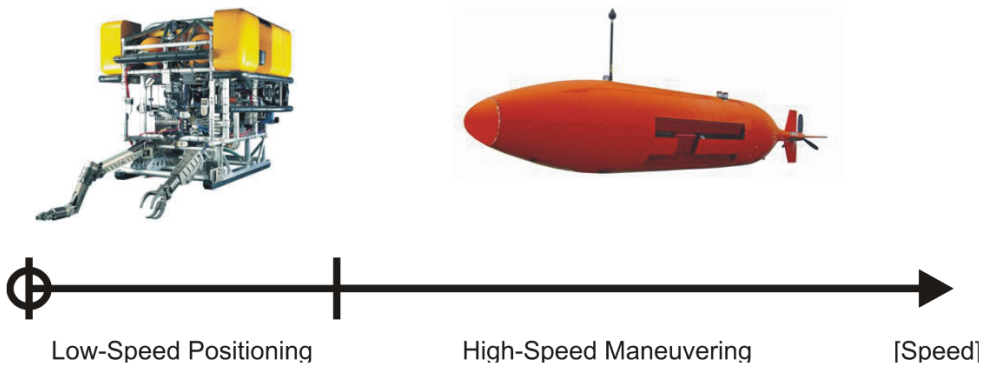

Figura 2.2: Um ROV (à esquerda), e um AUV (à direita). Adaptado de [24].

Em uma inspeção realizada por AUV/ROV, o veículo se dirige à estrutura de interesse e realiza a coleta de dados. Os ROVs e AUVs fazem parte de um grupo de veículos chamados de *unmanned underwater vehicles*, sendo que a principal diferença entre os dois se deve ao fato de que, enquanto o AUV

PUC-Rio - Certificação Digital Nº 2112316/CA

realiza a inspeção de forma independente, o ROV é conectado e operado a partir da superfície [25].

A escolha do veículo a ser utilizado depende do objetivo da inspeção. ROVs são capazes de transmitir dados em tempo real para a embarcação através do cordão umbilical, porém os AUVs geralmente apresentam maior velocidade e mobilidade [24]. Adicionalmente, ROVs podem ser de observação ou de intervenção, pois podem possuir braços mecânicos úteis para diversas tarefas, como mover objetos com potencial de danificar o duto para longe do mesmo ou usar um acessório como uma escova rotativa para realizar a limpeza do duto. Também é possível adaptar os sensores do ROV de acordo com o interesse, por exemplo, medir temperatura, salinidade ou potencial eletroquímico [26]. Logo, ROVs geralmente são utilizados em inspeções onde se necessita realizar algum tipo de intervenção, e AUVs naquelas onde o objetivo é mapear ou apenas avaliar visualmente os equipamentos [24].

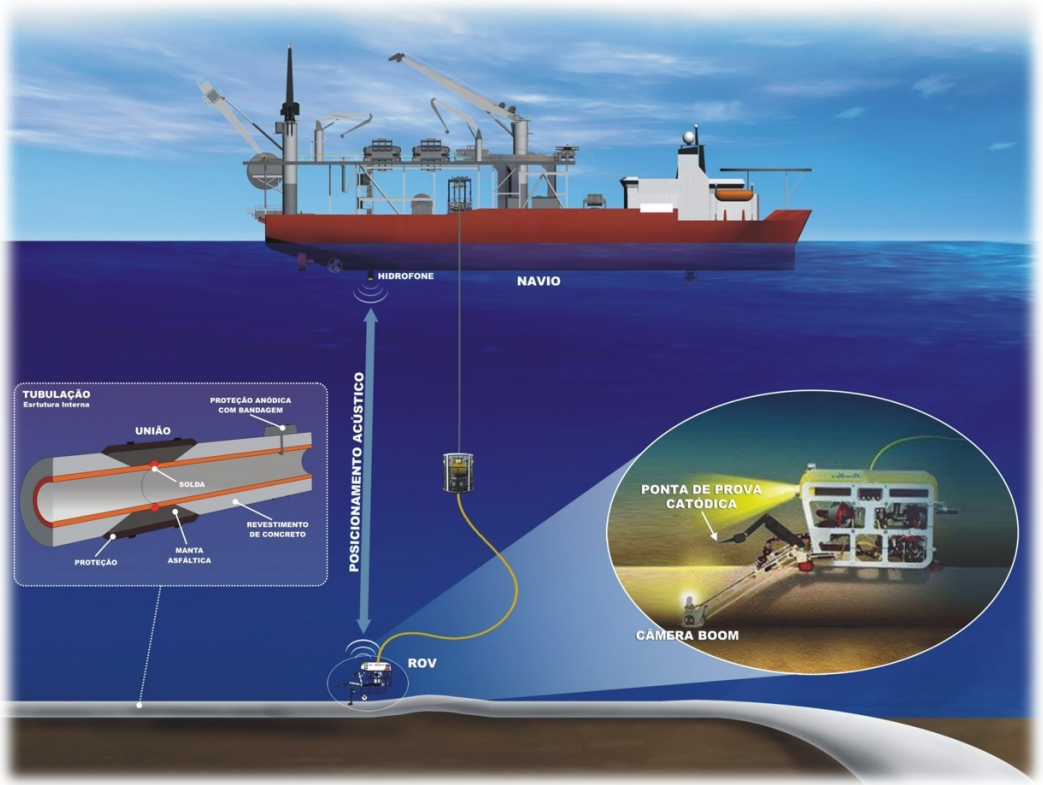

Figura 2.3: Esquemático de uma inspeção submarina realizada com foco no monitoramento de duto, com um ROV e seus equipamentos [27].

O foco do presente trabalho são as imagens de inspeções submarinas focadas no inspecionamento de dutos flexíveis e seus acessórios. Nesse contexto, as inspeções são em sua maioria realizadas por ROV, e as informações coletadas

PUC-Rio - Certificação Digital Nº 2112316/CA

são: as imagens geralmente através de duas ou mais câmeras, de forma sincronizada; um medidor de potencial de corrosão pode ser utilizado para medir no duto ou algum acessório; dados de mapeamento são coletados através de um sensor sonar; dados de posição do ROV de altitude com relação ao leito marinho; entre outros. Um exemplo de configuração de uma inspeção realizada para um tipo de duto é apresentada na Figura 2.3. Note que existe um cabo umbilical que liga o ROV ao navio durante todo o processo de inspecionamento, e a imagem também destaca uma ponta de prova catódica instalada no ROV, que é utilizada para verificar o estado de anodos de sacrifício que são instalados na tubulação. Adicionalmente, o ROV apresenta uma "câmera boom"que é um tipo de câmera lateral.

Após a coleta dos dados de interesse, um grupo de especialistas é responsável pela análise desses dados, avaliando a condição dos equipamentos inspecionados. Desde o processo de coleta de dados até a análise dos especialistas, existem normas nacionais e internacionais que regem o procedural que deve ser seguido. No caso de inspeções visuais realizadas por vídeo, o método requer que o especialista assista aos vídeos coletados pelo veículo submarino, buscando identificar qualquer forma de não-conformidade.

O resultado dessa inspeção visual realizada pelo especialista pode ser resumido em duas possibilidades: (1) as estruturas inspecionadas apresentam-se em condições ideais para funcionamento pleno; ou (2) algum tipo de irregularidade foi identificada, requerindo uma ação externa. Algumas irregularidades que podem ser observadas são: danos externos na capa do duto; enterramento do duto; danos em algum acessório; situação da proteção catódica irregular, entre outros. Todos os exemplos citados são potenciais fatores de desastres econômicos e ambientes, o que reforça a importância das inspeções submarinas.

PUC-Rio - Certificação Digital Nº 2112316/CA

2.1 Acessórios e Eventos de Inspeções Submarinas de Dutos Flexíveis

O transporte de óleo ou gás em ambiente submarino é realizado através de dutos, podendo estes serem flexíveis ou rígidos. A decisão sobre o tipo de duto a ser utilizado em um determinado projeto depende de diversos fatores, sendo que no Brasil existe uma hegemonia do tipo flexível. Portanto, a presente dissertação busca classificar acessórios e eventos de interesse presentes em imagens de inspeções submarinas coletadas no processo de inspecionamento de dutos flexíveis. Nessa seção serão apresentadas algumas estruturas, acessórios e ocorrências de interesse que virão a constituir as classes de interesse do problema de classificação.

A Figura 2.4 apresenta um exemplar de duto flexível. Esses dutos apresentam formato tubular e geralmente coloração amarela, sendo formados por diversas camadas internas, cada uma com uma função específica [28]. São utilizados em todo o interligamento existente entre equipamentos de extração, produção e escoamento de petróleo. A ligação entre dois dutos ou entre o encerramento de um duto é realizada por meio de um conector, chamado de *end fitting* (Figura 2.4.b). Um *end fitting* é formado por um corpo, uma seção de afinamento seguida por uma terminação, que geralmente é onde fica conectada uma flange (Figura 2.4.c). Esse acessório é responsável pela fixação entre dois *end fittings* no caso da conexão entre dois dutos; ou entre um único *end fitting* e um equipamento.

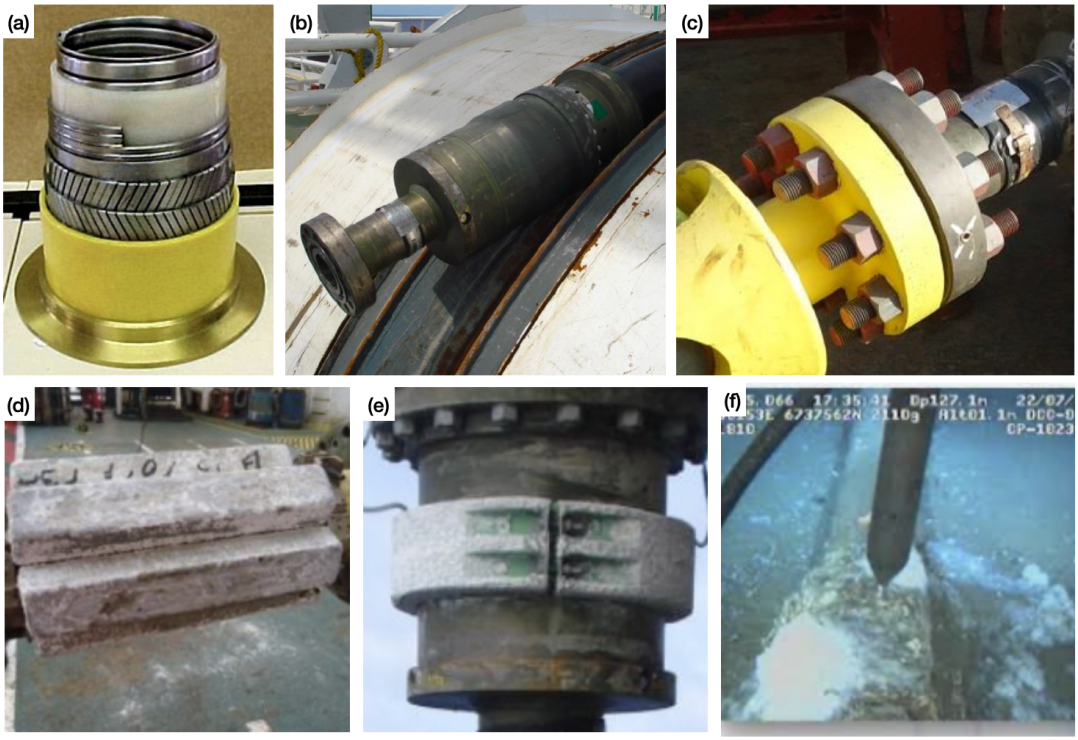

Figura 2.4: Em (a), a visão lateral de um duto flexível com suas camadas internas. Em (b), um exemplar de *end fitting* antes de sua instalação, provavelmente em uma embarcação. Em (c), uma flange. Dois tipos diferentes de anodo são apresentados, em (d) um anodo do tipo bracelete e (e) um anodo do tipo anel. Em (f), um anodo com sua proteção catódica comprometida [28, 29].

A salinidade do ambiente submarino onde esses dutos permanecem compreende um cenário de alta corrosidade eletroquímica. Nesse contexto, anodos têm sido uma solução amplamente utilizada na indústria, por constituirem uma proteção catódica segura e de baixo custo [30]. Esses anodos podem ser instalados no próprio duto ou em seus conectores, a depender do projeto. Na

PUC-Rio - Certificação Digital Nº 2112316/CA

Figura 2.4, observa-se dois tipos de anodos: um do tipo bracelete (2.4.d) e um do tipo anel (2.4.e), sendo que o exemplar do tipo anel está instalado em torno do corpo de um *end fitting*. Na Figura 2.4.f tem-se um anodo em processo de corrosão avançado, o que torna necessário a verificação do seu potencial eletroquímico de forma a conferir se a sua proteção catódica está comprometida.

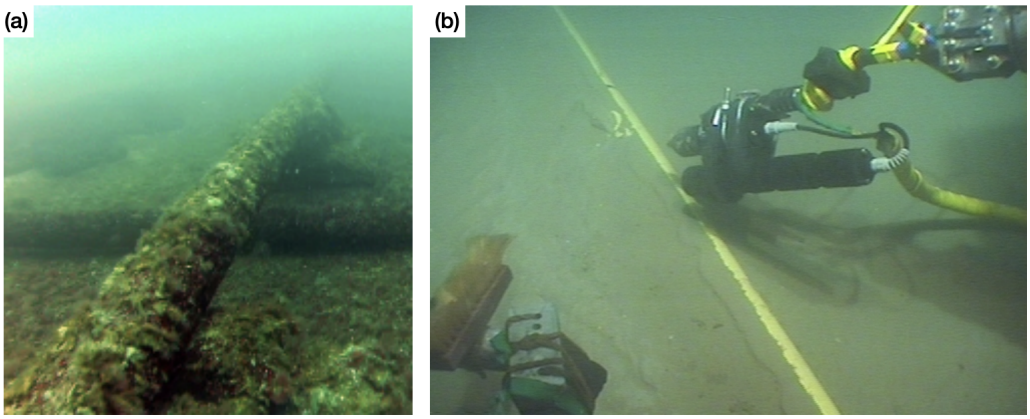

Figura 2.5: Em (a), um cruzamento entre dois dutos submarinos. Em (b), um duto parcialmente enterrado é inspecionado por um ROV através de um medidor [31, 32].

PUC-Rio - Certificação Digital Nº 2112316/CA

O cruzamento entre dutos e o assoreamento de dutos são dois eventos que devem ser acompanhados, pois podem apresentar complicações (Figura 2.5). Quando há o cruzamento de dutos (Figura 2.5.a), por exemplo, pode ocorrer que algum objeto localizado em solo marinho, como uma sucata metálica, fique preso na região onde os dutos se cruzam, o que pode vir a danificar ambos os dutos. Por outro lado, o duto se encontrar completamente enterrado pode conferir um risco à sua integridade, visto que a pressão exercida pelo solo marinho pode contribuir para sua degradação com anos de uso (Figura 2.5.b).

A Figura 2.6.a apresenta um outro tipo de ocorrência de interesse durante uma inspeção, que consiste na medição do potencial eletroquímico de algum acessório do duto, realizada pelo medidor do ROV. Adicionalemente, um outro acessório que pode aparecer em parte da tubulação é o flutuador, representado na Figura 2.6.b. Esse acessório é geralmente encontrado em *risers* flexíveis, que são tubulações que ligam uma estrutura flutuante, como por exemplo uma plataforma, ao restante do sistema submarino (Figura 2.7). O intuito da instalação desses flutuadores é a redução do esforço mecânico presente nas conexões entre dutos e/ou entre dutos e outras estruturas submarinas, atuando como amortecedores.

Adicionalmente, existem outros objetos auxiliares que são utilizados na instalação de tubulações submarinas, como cordas e amarras (Figura 2.6.c

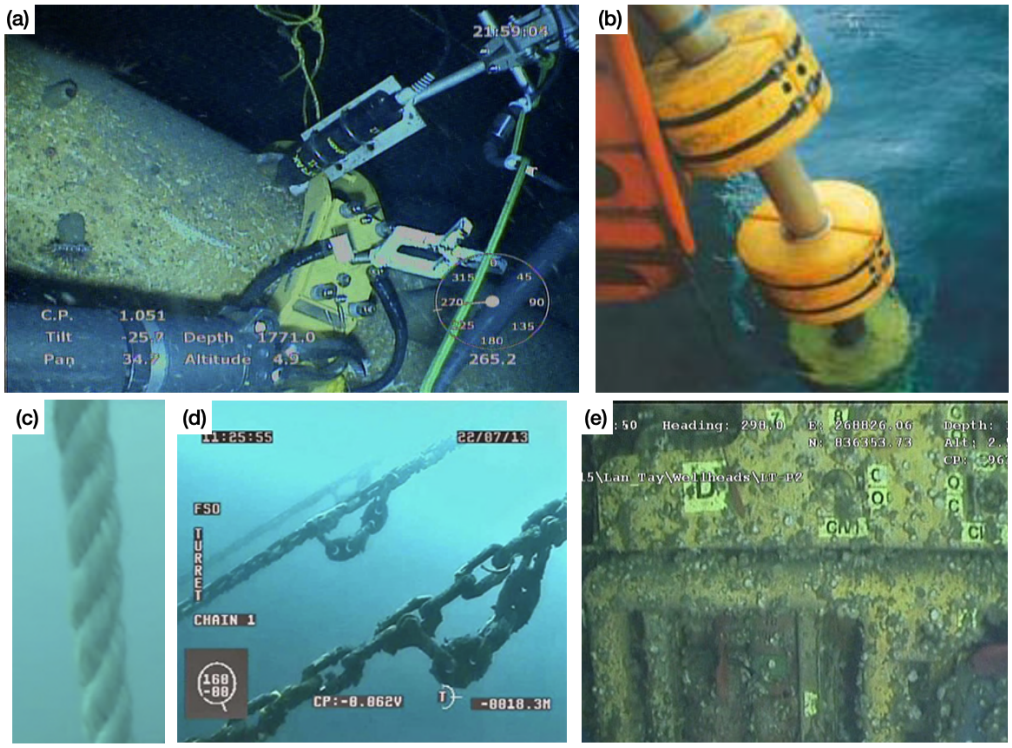

Figura 2.6: Em (a), uma imagem que apresenta o braço robótico do ROV e um medidor também manipulado pelo ROV inspecionando um acessório no duto. Em (b), flutuadores no processo de imersão de uma tubulação. Em (c), uma corda. Em (d), uma sequência de amarras. Em (e), um equipamento com texto em sua superfície externa [31, 33, 34, 35, 36].

e 2.6.d). As cordas podem ser utilizadas em diversas situações em que seja necessário prender algum objeto ao duto, ou também no processo de submersão do mesmo, onde cordas podem ser utilizadas para o lançamento do duto. Já as amarras, geralmente aparecem em situações onde flutuadores estão conectados a algum tipo de colar no duto em solo submarino. Por último, outro evento de interesse são textos que estão presentes em dutos ou equipamentos (Figura 2.6.e).

Um outro evento de interesse é a presença de sucata próxima ao duto. Em 2020, estimou-se que existiam $2000 kg/km^2$ de lixo no fundo do mar [37]. Além dos desafios que a grande quantidade de resíduos no solo submarino confira à biodiversidade oceânica, essa sucata também pode danificar dutos, seus acessórios ou equipamentos. A periculosidade depende do tamanho e da composição da sucata, sendo as metálicas e/ou maiores as que apresentam mais risco, e portante não é interessante que esses objetos fiquem próximos ou em contato direto com dutos.

Por último, também é de interesse acompanhar a presença de vida marinha na região das instalações dos equipamentos e tubulações. Nesse

PUC-Rio - Certificação Digital Nº 2112316/CA

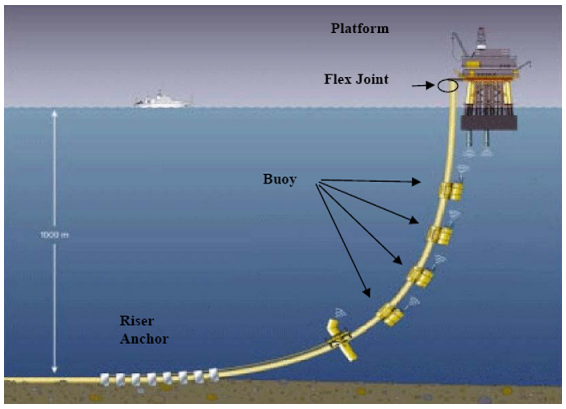

Figura 2.7: Exemplo de uma instalação de *risers* com flutuadores [38]

.

contexto, animais e outros seres vivos podem vir a danificar as tubulações ou equipamentos. Por exemplo, um crescimento exacerbado de corais pode vir a ser causa da corrosão de algumas tubulações, o que deve ser corrigido quando identificado.

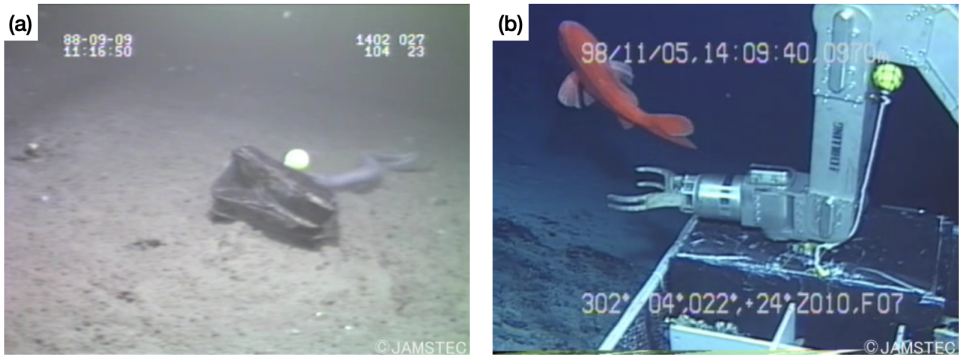

Figura 2.8: Em (a), uma sucata em solo marinho. Em (b), um peixe próximo ao braço mecânico de um ROV [39].

PUC-Rio - Certificação Digital Nº 2112316/CA

3
Referencial Teórico

Nesse capítulo são expostos os fundamentos teóricos relacionados ao tema da presente dissertação. Primeiramente, é realizada uma introdução aos diferentes paradigmas existentes na literatura de Aprendizado de Máquina na seção 3.1. Em seguida, na seção 3.2, é introduzido o problema de classificação de imagens e as métricas de desempenho utilizadas para aplicação escolhida. Na seção 3.3 são apresentadas as arquiteturas na qual os modelos utilizados são baseados. Por último, é realizada uma revisão de trabalhos relacionados ao presente trabalho.

3.1
Paradigmas de Aprendizado de Máquina

A área de estudo conhecida como Aprendizado de Máquina, ou *Machine Learning* (ML), compreende um campo da Inteligência Artificial que foca no desenvolvimento de agentes inteligentes capazes de adquirir conhecimento através de um processo de aprendizagem, tendo como objetivo a resolução de uma tarefa [40]. Tradicionalmente, esse processo de aprendizagem é composto (1) uma etapa chamada de treinamento, na qual o agente é exposto a um conjunto de dados, adiquirindo experiência e aprendendo algo sobre os mesmos. e (2) pelo menos uma etapa de validação na qual o conhecimento adiquirido é avaliado, por meio de medidas numéricas da performance dos modelos. Após essas etapas, espera-se que o modelo aprendido apresente performance satisfatória, produzindo a saída correta [41].

Tipicamente, métodos de ML são agrupados em duas categorias macro, de acordo com o nível de supervisão utilizado na etapa de treinamento: aprendizado supervisionado e não-supervisionado [42]. Nesse contexto, supervisão significa o acesso a rótulos durante o treino, de forma que o conjunto de dados seja formado por pares entrada-rótulo [43]. Nos últimos anos, a pesquisa de aprendizado não-supervisionado evoluiu para algumas áreas adjacentes, dentre as quais encontram-se os métodos de aprendizado semi-supervisionado e auto-supervisionado [44]. Nessa sessão, será realizada uma introdução a três tipos de aprendizado que serão citados ao longo deste trabalho: aprendizado supervisionado, semi-supervisionado e auto-supervisionado.

PUC-Rio - Certificação Digital Nº 2112316/CA

3.1.1 Aprendizado Supervisionado

Métodos de Aprendizado Supervisionado são aqueles onde o conjunto de dados utilizado durante o treinamento é completamente rotulado, e a tarefa de aprendizagem consiste no mapeamento de sinais de entrada a sinais correspondentes de saída [45]. Esses sinais de saída podem ser utilizados como uma variável objetivo durante a otimização do modelo no caso de problemas de regressão; ou como a variável de categorização de dados em problemas de classificação. Após o processo de aprendizagem, esses modelos são utilizados em tarefas de predição, onde se busca prever uma saída dado um sinal de entrada previamente desconhecido.

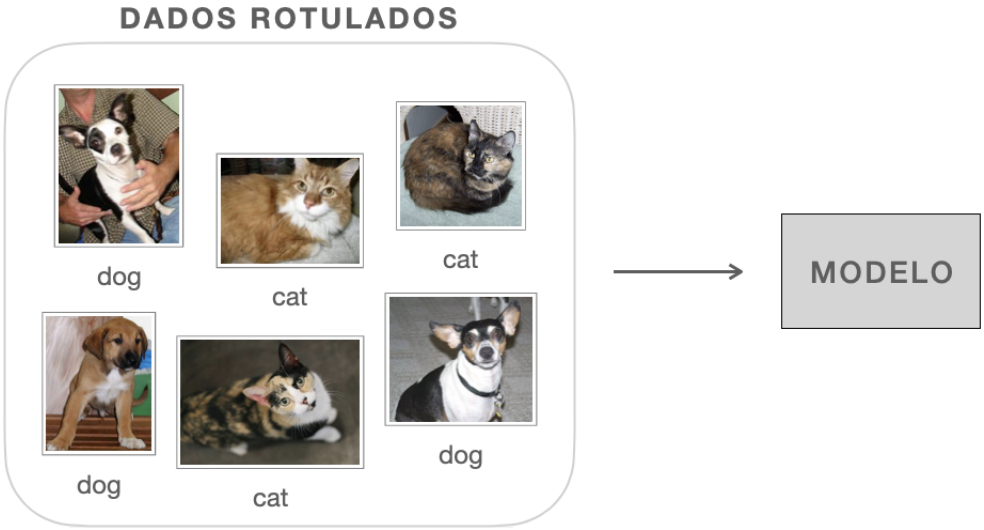

Figura 3.1: Aprendizado supervisionado.

3.1.2 Aprendizado Auto-supervisionado

A classe de métodos chamada de Aprendizado Auto-supervisionado é composta por modelos capazes de extrair informações de um conjunto não-rotulado, tendo em vista utilizá-las como "rótulos" de suas amostras originárias durante o processo de aprendizado [46]. Logo, tais métodos herdam seu nome pela capacidade de extrair um "par" de entrada-saída – como utilizado no processo de treinamento de métodos supervisionados – a partir de um único dado não-rotulado de entrada. Essas informações podem ser de diversos tipos e são aproveitadas na etapa de pré-treino, que pode ou não ser seguida por uma etapa de *fine-tuning*. No contexto de visão computacional, os "rótulos"

PUC-Rio - Certificação Digital Nº 2112316/CA

extraidos podem ser, por exemplo, diferentes resultados obtidos para uma mesma imagem quando aplicada uma transformação geométrica.

Procedimentos experimentais de aprendizado auto-supervisionado definem uma *pretext task* que é uma tarefa a ser aprendida o pré-treino, e uma *downstream task*, que será realizada pelo modelo em produção, após a etapa de *fine-tuning* – quando essa última for aplicável. A *pretext task* é definida buscando que o modelo seja capaz de aprender a extrair representações úteis dos dados de entrada, projetadas em um espaço latente. Idealmente, essas representações serão proveitosas na aplicação final – a *downstream task* – que pode ser uma tarefa de classificação, por exemplo. Essa ideia de aprendizado envolvendo um processo de aprendizado (ou uma rede dedicada) para extração de representações – ou *features* – e uma segunda parte/rede para tarefa final, costuma ser chamado na literatura de *representation learning*.

Seguindo o exemplo de aplicar uma transformação na imagem para gerar os pares de dados, a *pretext task* a ser considerada durante o pré-treino poderia ser definida como o modelo tentar prever se dois resultados – de imagens transformadas – pertencem a uma mesma imagem. Após essa etapa, o modelo poderia passar por uma etapa de *fine-tuning*, onde aproveitaria a experiência do pré-treino e aprenderia uma *downstream task* de classificação de imagens [47], detecção de objetos [48], entre outras aplicações.

3.1.3 Aprendizado Semi-supervisionado

Os métodos agrupados como Semi-supervisionados realizam a etapa de treinamento utilizando dados cujos rótulos estão disponíveis apenas para uma parcela do conjunto. Essa família de métodos pode ser enxergada como uma combinação dos paradigmas de Aprendizado Supervisionado e Não-supervisionado, pois pode empregar conceitos do primeiro para a parte rotulada do conjunto, e do segundo para a parte não-rotulada [45].

Modelos desse tipo assumem que existe uma relação entre a distribuição dos dados observada no campo dos rótulos (para a parte rotulada do conjunto de dados) e a distribuição observada em todos os dados de entrada – rotulados e não-rotulados. Partindo desta premissa, tais modelos buscam extrair conhecimento utilizando todos os dados disponíveis para um dado problema [49]. Essa capacidade de utilizar dados não-rotulados em conjunto com os rotulados expande as possibilidades de aplicações desses modelos em indústrias onde rótulos são altamente custosos e/ou escassos [50].

PUC-Rio - Certificação Digital Nº 2112316/CA

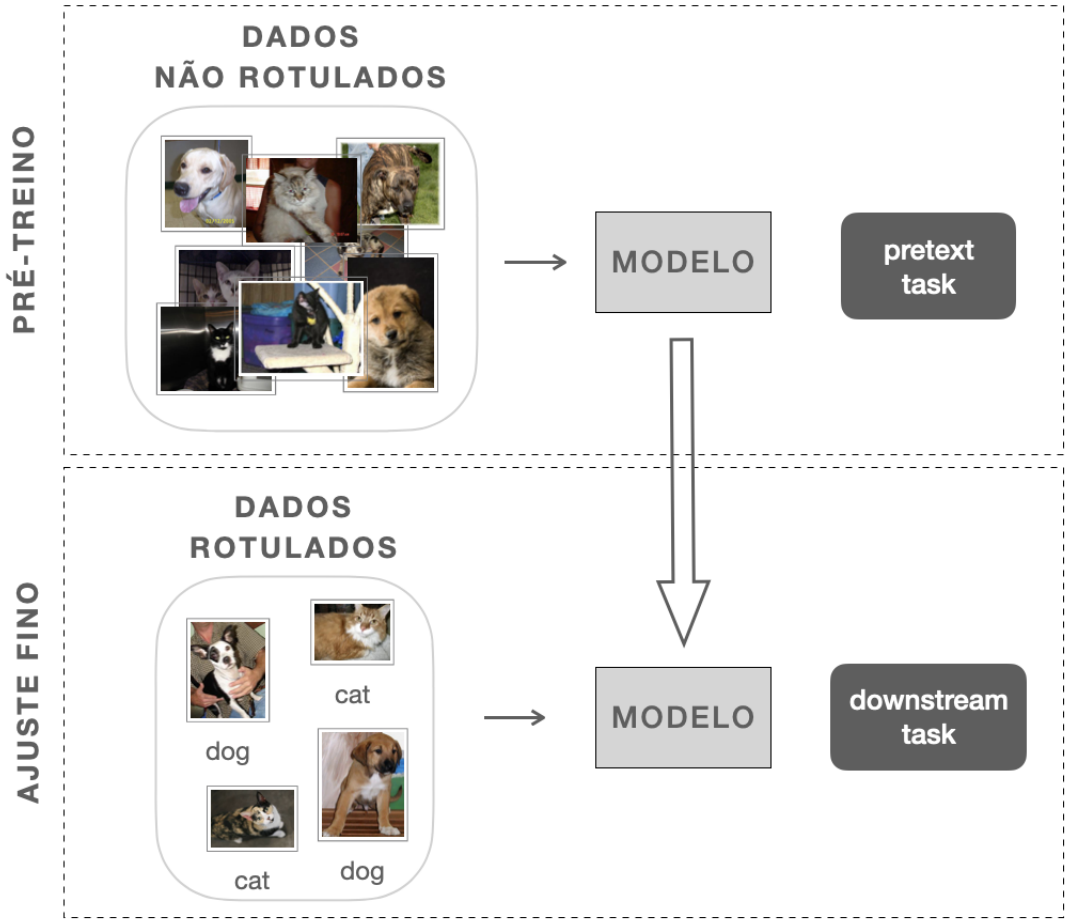

Figura 3.2: Aprendizado Auto-supervisionado.

3.2 Classificação de Imagens

O problema de classificação é um dos principais desafios da área de Aprendizado de Máquina, constituindo tópico igualmente relevante para aplicações de Visão Computacional. Seja X um conjunto de dados de tamanho N, onde cada amostra é denotada por x_i, onde $i = 1, 2...N$. Cada ponto amostral x_i possui um vetor de label(s) correspondente y_i, proveniente da matriz de labels Y. O vetor y_i é uma codificação da(s) classe(s) C_k à(s) qual(is) x_i pertence, onde $k = 1, 2...K$, sendo K o número de classes. A tarefa de classificação consiste em tentar prever as classes presentes em y a partir de valores amostrados de x [43].

Em um contexto de classificação de imagens, X representa um conjunto onde cada ponto x_i é uma imagem ou uma ou mais de suas *features*; e o objetivo do aprendizado é que o modelo seja capaz de determinar a qual categoria(s) os objetos presentes na imagem x_i pertence, ou seja, que seja capaz de gerar um

PUC-Rio - Certificação Digital Nº 2112316/CA

Figura 3.3: Aprendizado Semi-supervisionado.

vetor de saída $\hat{y}_i$ que se aproxime de y_i em um nível satisfatório. Métodos de classificação de imagens podem ser agrupados de acordo com a quantidade de classes que integram o problema e que possam estar presentes em uma mesma amostra: métodos de classificação binária, multi-classe e *multi-label*.

Problemas de classificação binária compreendem aqueles onde há apenas duas classes, ou seja, $K = 2$. Por exemplo, o treinamento de um modelo detector da presença (primeira classe) ou ausência (segunda classe) de um determinado objeto em uma imagem constitui um problema de classificação binária. Nesse caso, a abordagem comumente usada considera o vetor y como sendo unitário e binário, podendo apresentar valor 1 (amostra pertence à classe C_1) ou 0 (classe C_2) [43]. Por outro lado, quando se torna necessária uma abordagem que envolva três ou mais classes para desenvolvimento de uma solução, os modelos competentes para realizar tal tarefa são os de classificação multi-classe [51]. Modelos que fazem parte desse grupo são aptos para a resolução de problemas que lidam com conjuntos de dados que envolvam até milhares de classes, como é o caso dos modelos *benchmark* do conjunto de imagens público ImageNet [13].

Formalmente, em métodos multi-classe, o modelo aprendido é capaz de mapear uma entrada x para o vetor binário y, que apresenta K posições. O valor 1 ou 0 é atribuído a cada posição k de y, e representa a presença (1) ou ausência (0) da k-ésima classe C_k. Nesse contexto, os valores de cada posição k do vetor predito $\hat{y}_i$ podem ser interpretados como a probabilidade (p_k) que a classe (C_k) esteja presente na imagem x_i [5]. Em problemas desse tipo, as classes aprendidas são mutuamente exclusivas. Na prática, isso significa que o vetor y deve apresenta valor 1 para apenas uma de suas posições, enquanto as demais permanecem zeradas [43]. Isso gera a limitação de que a soma $somatoriop_k$ deve ser limitada a 1 durante o aprendizado [52].

PUC-Rio - Certificação Digital Nº 2112316/CA

A classificação multi-label de imagens pode ser visualizada como uma generalização do método multi-classe introduzido acima. Diferente do cenário multi-classe tradicional, em conjuntos *multi-label* as categorias são não-exclusivas. Isso significa que uma imagem x pode estar associada a mais de uma classe C_k. Em outros termos, em aplicações *multi-label*, o vetor de saída $\hat{y}_i$ gerado pelo modelo a partir de uma entrada x_i pode apresentar valor 1 para uma ou mais posições, representando a presença de uma ou mais classes em x_i. Nesse sentido, o vetor predito $\hat{y}$ pode apresentar soma $somatoriop_k$ maior do que [53].

Durante o treinamento do modelo, geralmente é realizada uma avaliação de sua performance ao final de cada época e/ou ao final do treino. Dessa forma, é possível selecionar o modelo de melhor desempenho para a aplicação desejado. Para que seja possível selecionar um modelo de maneira objetiva, utilizam-se métricas de desempenho.

3.2.1 Métricas de Desempenho

Métricas de desempenho são medidas numéricas da performance de modelos. Essas medidas geralmente são calculadas em cima de uma partição de validação ou de teste. A escolha de uma métrica para avaliar o modelo deve considerar diversos fatores que envolvem o problema, como o número de classes e a distribuição das mesmas, isto é, se o conjunto de dados está devidamente balanceado. Adicionalmente, em aplicações na indústria, também pode ser relevante realizar uma análise de quais deverão ser os erros de maior custo.

A avaliação de modelos multi-label difere do cenário tradicional – binário ou multi-classe – devido à maior complexidade do processo de otimização dos modelos, pelo maior grau de liberdade no que diz respeito às classes presentes em uma amostra. Logo, torna-se necessário a adaptação das métricas pelo meio de métodos de regularização e/ou ponderação, de forma que seja possível realizar uma avaliação de cada classe com diligência.

Nessa sessão, serão definidas as métricas a serem utilizadas para avaliar e comparar a performance dos modelos testados nesse trabalho. Primeiramente, é realizada uma introdução a conceitos fundamentais para compreensão do cálculo das métricas. Na sequência, as métricas são apresentadas.

3.2.1.1 Definições

Na literatura, o conceito de *True Positives* (TP) diz respeito ao número de amostras corretamente classificadas como pertencente a uma determinada

PUC-Rio - Certificação Digital Nº 2112316/CA

classe. Já o chamado *False Positives* (FP) se refere ao número de casos incorretamente classificados como pertencente à classe. Adicionalmente, *True Negative* (TN) é o número de casos corretamente classificados como não pertencente, e *False Negative* (FN) o número de casos incorretamente classificados como não pertencente à classe avaliada [54].

3.2.1.2 Precisão

Precisão é uma métrica que mede o número de *True Positives* entre as amostras classificadas como pertencentes a determinada classe [54].

$$\text{Precisão} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3-1}$$

3.2.1.3 *Recall*

Também chamado de Sensibilidade ou *True Positives Rate* (TPR), *Recall* é a medida de *True Positives*, isto é, o número de amostras classificados como pertencentes a determinada classe, entre todas as amostras originalmente pertencentes à esta [54].

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3-2}$$

3.2.1.4 *F1-score*

O *F1-score* é uma média harmônica ponderada da Precisão e *Recall*:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}} \tag{3-3}$$

Um bom *F1-Score* significa um baixo número de *False Positives* e *False Negatives* [54]. Essa medida é usada quando se busca um equilíbrio entre precisão e recall, e também quando a base de dados em questão é desbalanceada, podendo apresentar um alto número de amostras negativas (*True Negatives*) ao avaliarmos cada classe separadamente.

PUC-Rio - Certificação Digital Nº 2112316/CA

3.3 Redes Neurais Artificiais para Classificação de Imagens

Nessa seção, será realizada uma revisão da literatura de Redes Neurais Artificias utilizadas em problemas de Classificação de Imagens.

3.3.1 Introdução: do Perceptron ao Aprendizado Profundo

Redes Neurais Artificiais (RNA) são um conjunto de modelos da área de ML, inicialmente formulados a partir do modo pelo qual se acreditava que o cérebro realiza suas tarefas [55]. Essa inspiração neurológica partiu de um trabalho publicado por Frank Rosenblatt em 1958, onde apresenta o seu modelo matemático de um neurônio biológico, chamado de *Perceptron* [56]. O modelo proposto pode ser interpretado como uma rede neural de apenas um *neurônio* – como são chamados os nós das redes neurais –, com função de ativação linear, e pesos adaptáveis. Adicionalmente, existe um viés que é somado à saída do neurônio.

PUC-Rio - Certificação Digital Nº 2112316/CA

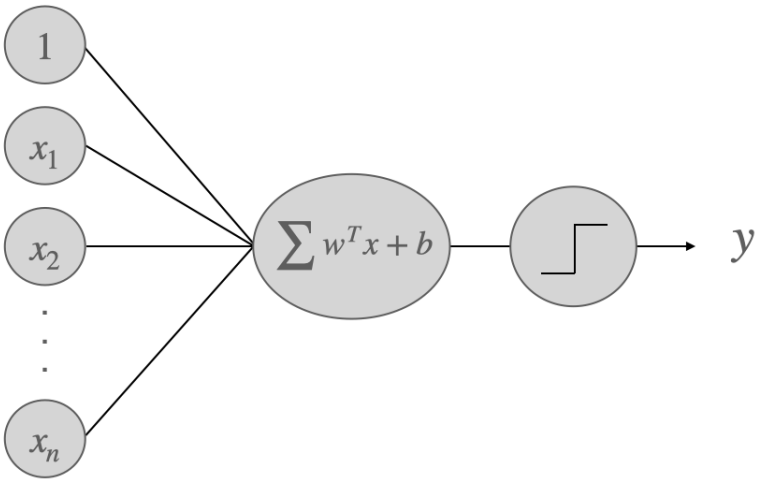

Figura 3.4: Modelo perceprtron de Rosenblatt.

Por apresentar apenas uma camada linear, o *Perceptron* só é capaz de resolver problemas linearmente separáveis.Pela necessidade de lidar com problemas não-lineares surgem as RNAs mais complexas, a partir do chamado *Multi-layer Perceptron* (MLP) [43]. Cada camada do MLP é formada por vários neurônios, sendo a saída de uma a entrada de outra, até a última camada da rede. As primeira camada é chamada de camada de entrada, a última é chamada de camada de saída e as intermediárias são chamadas de camadas ocultas. A principal diferença entre o MLP e o modelo *Perceptron* é a existência da não-linearidade, pela introdução de funções de ativação

sigmoidais nas camadas ocultas. Adicionalmente, no processo de treinamento do MLP é introduzido o algoritmo da retro-propagação do gradiente do erro, utilizado no treinamento de redes profundas mais modernas [7].

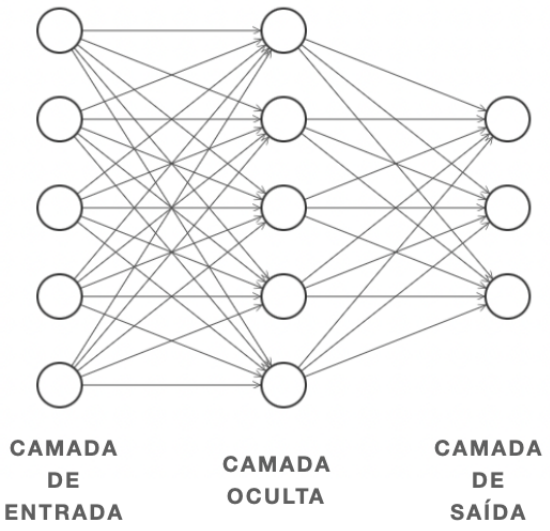

Figura 3.5: Multi-layer Perceptron.

PUC-Rio - Certificação Digital Nº 2112316/CA

Com o advento da computação de alto desempenho nos anos 2000 com a implementação de redes neurais para *Graphics Processing Units* (GPUs), surge novamente um interesse na área e as aplicações em visão computacional germinam [57]. Desde 2009, as chamadas *Redes Neurais Profundas* – da área que ficou conhecida como *Deep Learning*, ou Aprendizado Profundo – têm mantido estavelmente o primeiro lugar em *benchmarkings* internacionais de desafios de reconhecimento de padrões [58]. Essas redes recebem esse nome por contarem com milhares de neurônios, distribuídos em várias camadas. Dentre as redes profundas, as redes neurais convolucionais têm sido a de maior relevância para problemas de classificação de imagem, conquistando posições altas em benchmarks da área. Com a introdução dos modelos do tipo Vision Transformers, estes se consolidaram como atual estado-da-arte em problemas de classificação.

3.3.2 Redes Neurais Convolucionais para Classificação de Imagem

Redes Neurais Convolucionais, ou *Convolutional Neural Networks* (CNNs), constituem um grupo de redes capazes de processar dados que apresentem uma estrutura em *grid*, através de suas camadas convolucionais [7], de onde herdam seu nome. Essas redes são formadas, em resumo, por um

empilhamento de camadas convolucionais, camads de *pooling* e camadas de ativação. De acordo com a arquitetura proposta, essas camadas podem ser completamente conectadas, isto é, com todos os neurônios de uma conectadas a outra, ou não. Entre cada neurônio conectado, há um peso que é otimizado durante o treinamento. Através do processo de treinamento, espera-se que a rede aprenda – por meio do ajuste dos pesos entre seus neurônios – a gerar uma saída adequada dado um sinal recebido em sua camada de entrada.

No artigo que introduz a LeNet [59], os autores propoem uma CNN aplicada ao reconhecimento de dígitos. Nessa contribuição, uma das questões discutidas é o processo de treinamento das CNNs, que é baseado na retro-propagação do gradiente do erro calculado a partir da comparação da saída esperada e a saída obtida para um sinal na entrada da rede. Em 2012, a arquitetura conhecida como AlexNet [60] é proposta, tornando-se a primeira solução baseada em CNN a conquistar o primeiro lugar do desafio de classificação de imagens do conjunto ImageNet (*ImageNet Large Scale Visual Recognition Challenge*, ou ILSVRC) [61]. Os autores introduzem a camada de ativação ReLU e utilizam camadas drop-out para tentar superar o problema de sobreajuste dos dados.

Os autores da VGGNet [62] avaliam o impacto da profundidade de uma rede em sua performance final. Nesse contexto de CNNs, profundidade refere-se ao número de camadas convolucionais presente na arquitetura de uma rede. Dentre as configurações testadas, com 11, 13, 16 e 19 camadas, a que obteve maior êxito na tarefa avaliada foi a com 16 camadas. Isso indica que existe um ponto de saturação no ganho em performance obtido através do aumento da profundidade da rede. Os autores da GoogLeNet [63] mantêm a ideia de utilizar uma rede mais profunda, e introduzem o módulo *Inception* em sua arquitetura, conquistando o primeiro lugar do ILSVRC.

Nos últimos anos, as redes CNNs têm se mantido dentre os tipos de rede mais utilizados em aplicações de classificação de imagens, devido à sua alta capacidade de extração de atributos e performance satisfatória em tais aplicações. A cada mês surge uma grande variedade de novos artigos resolvendo problemas de visão computacional a partir da aplicação de uma rede desse tipo.

3.3.2.1 Camada Convolucional

A camada convolucional é o pilar central das CNNs. Essas camadas utilizam filtros (ou *kernels*), de tamanho e passo (ou *stride*) variáveis, para realizar operações de convolução em uma imagem de entrada. A saída obtida após a camada convolucional é comumente chamada de *feature map* (mapa de atributos) ou *activation map* (mapa de ativação). A operação de convolução

PUC-Rio - Certificação Digital Nº 2112316/CA

consiste em arrastar o filtro pela imagem de entrada, calculando o produto interno entre o filtro e a imagem para cada posição espacial. A operação está representada na Figura [citar], onde uma imagem de entrada de dimensões $HXWXD$ é convolucionada por um filtro de dimensões $hxwxd$ com *stride* s. Para cada posição i e j, é calculado o produto interno daquela porção da imagem com o filtro, gerando o mapa de ativação de dimensõe $(H-2)X(W-2)X1$.

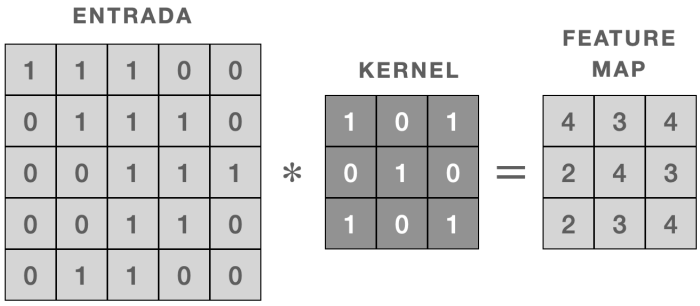

Figura 3.6: Operação de convolução, com uma imagem 5x5, filtro de tamanho 3x3 e *stride* 2.

PUC-Rio - Certificação Digital Nº 2112316/CA

3.3.2.2 Camada de Pooling

A cama de *pooling* aplica uma operação de *downsampling* na imagem recebida na sua entrada, realizando a redução da resolução original. Estatisticamente, pode-se afirmar que a essa camada permeia o papel de tornar o mapa de ativação menos sensível a pequenas variações na entrada, visto que um ponto da saída gerada consiste em um resumo das *features* presentes na área abrangida durante o *downsampling*. Esse resumo pode ser a média, no caso das camadas de *average pooling*, ou o valor máximo para camadas de *max pooling*.

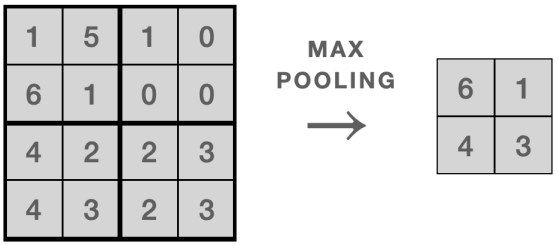

Figura 3.7: Operação de *pooling*, com um filtro de tamanho 2x2 e *stride* 2.

3.3.3
Vision Transformers: Modelos Transformers aplicados à Visão Computacional

Modelos do tipo Transformers constituem um grupo de modelos baseados em módulos de atenção. Esses módulos de atenção permitem que as redes selecionem a parte da entrada que seja mais relevante para gerar uma saída, aumentando a eficiência da rede [64]. Em redes mais tradicionais como as convolucionais ou as recorrentes, não há a possibilidade de paralelização do processamento dos dados de entrada, o que dificulta o aprendizado da relação entre dois pontos distantes. Para essas redes, como os dados são processados sempre de forma sequencial, o custo computacional cresce de forma proporcional ao volume de dados.

A grande vantagem de usar uma arquitetura com mecanismos de atenção ao invés das mais tradicionais redes convolucionais ou as redes recorrentes é a simplicidade e a eficiência proposta. Por utilizar o módulo de atenção, o modelo dispensa a necessidade de processar toda a matriz de entrada em ordem sequencial, o que permite maior paralelização do processamento. Logo, o processamento da entrada é limitado a um número fixo de operações, o que facilita o aprendizado de relações entre dois pontos de entrada arbitrários.

O atual estado-da-arte de módulos de atenção, conhecido da literatura como *self-attention* ou *intra-attention* [64], é o utilizado pelos Transformers. Nesse mecanismo, a representação de uma sequência de entrada é gerada a partir da *relação* entre diferentes pontos dessa sequência. Seja uma entrada $\mathbf{X}$ e sua respectiva saída $\mathbf{Y}$. Inicialmente, $\mathbf{X}$ é projetada em três matrizes, uma de *query*, uma *key* e uma *value*. Para cada valor $\mathbf{v}$ da entrada, a saída é uma soma ponderada com pesos A_{ij}. Esses pesos são calculados com base na semelhança entre dois itens da sequência e suas respectivas projeções *query* $\mathbf{q^i}$ e *key* $\mathbf{k^j}$. A semelhança entre dois itens é calculada como sendo a soma dos valores do produto interno entre os dois vetores, seguidos por uma softmax.

3.3.3.1
Vision Transformers

Os chamados Vision Transformers (ViTs) são modelos baseados no processamento de imagens de entrada como uma sequência, análogo ao que é feito para dados textuais pelos Transformers originais do NLP. ViTs são o estado-da-arte atual em termos de precisão e eficiência em tarefas de classificação de imagens [18]. Inicialmente, cada imagem é transformada em uma sequência de *patches*, que são pequenos recortes da imagem. Então, cada *patch* é concatenado a uma identificação posicional (*position embeddings*)

PUC-Rio - Certificação Digital Nº 2112316/CA

que designa a posição do *patch* tendo referência a imagem original. Além disso, um *token* de classificação como o utilizado no modelo BERT [65] é adicionado à sequência de entrada, e o treinamento é feito de maneira totalmente supervisionada. O estágio de pré-treinamento é tradicionalmente feito com um perceptron multicamada como cabeça de classificação, que pode ser substituído por uma camada densa em uma etapa de ajuste fino.

A arquitetura de um ViT é resumida por um modelo *encoder* f_θ e uma cabeça de classificação formada por um MLP (Figura 3.8). Dentro da rede *encoder*, temos dois módulos principais: um módulo de *multi-headed attention* e um MLP, ambas precedidas por uma camada de LayerNorm (representada por "Norm"na Figura 3.8). Os módulos de *multi-headed self-attention* (ou apenas *multi-headed attention*) são o mecanismo responsável pela capacidade dos Vision Transformers de traçar dependências globais entre sinais de entrada e sinais de saída. Esse mecanismo *multi-headed* consiste em uma extensão dos módulos de atenção *self-attention* tradicionais, na qual a diferença está na utilização de diversas *heads* de atenção. Em outras palavras, k operações de *self-attention* são executadas em paralelo.

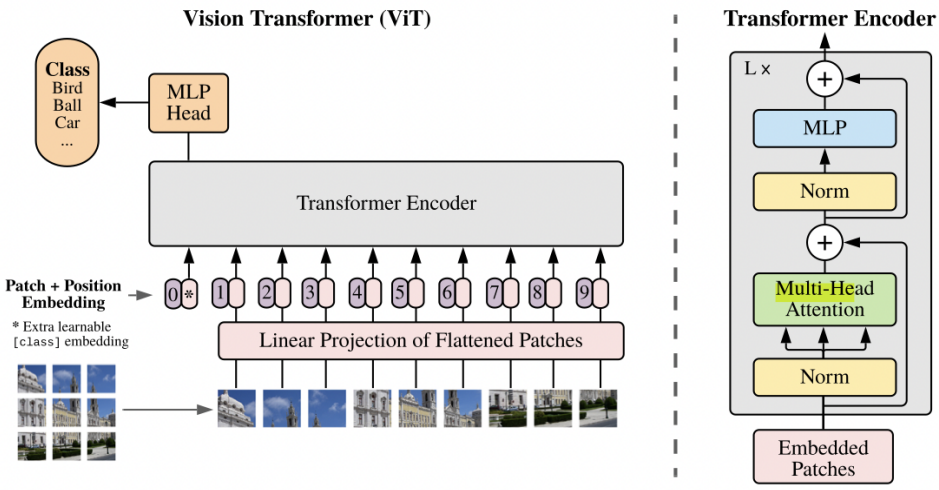

Figura 3.8: Arquitetura de um Vision Transformer [18].

3.4 Trabalhos Correlatos

3.4.1 ResNet-50

ResNet é um acrônimo para *Residual Network*, recebendo esse nome pela presença dos blocos residuais introduzidos pelos autores [66]. Já o número 50 da

PUC-Rio - Certificação Digital Nº 2112316/CA

ResNet-50 remete ao número de camadas presente nessa CNN, sendo formada por 48 camadas convolucionais, uma camada de *max pooling* e uma camada de *average pooling*. O bloco residual introduzido pelos autores busca solucionar o problema de *vanishing gradient* observado em redes muito profundas, que dificulta o treinamento e faz com que a performance final seja inferior. Os blocos residuais fazem com que o modelo aprenda uma distribuição residual. Ou seja, ao invés do modelo ser capaz de aprender apenas uma distribuição direta $H(\mathbf{x})$, esses blocos permitem que a rede aprenda uma distribuição $F(\mathbf{x}) = H(\mathbf{x}) - \mathbf{x}$, e o mapeamento original é reajustado para $F(\mathbf{x}) = H(\mathbf{x}) + \mathbf{x}$ (Figura 3.9). Os autores indicam ser mais fácil aprender essa distribuição ajustada [66].

A ideia de aprendizagem de uma representação residual se torna possível pela utilização de *skip connections* na arquitetura. Essas consistem em conexões que pulam uma ou mais camadas, possibilitando que o modelo aprenda uma função identidade ao somar a saída de camadas anteriores diretamente à saída de uma camada posterior. Cada bloco residual é formado por duas camadas convolucionais com uma função de ativação ReLU [67] e uma *skip connection* que soma a entrada da primeira camada à aída do bloco. A arquitetura completa da ResNet-50 consiste em um empilhamento de diversos blocos residuais, conforme indicado na Figura 3.10.

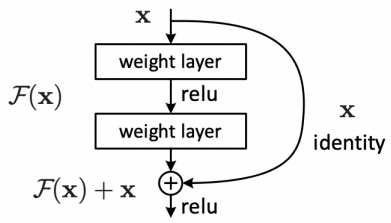

Figura 3.9: Bloco residual.

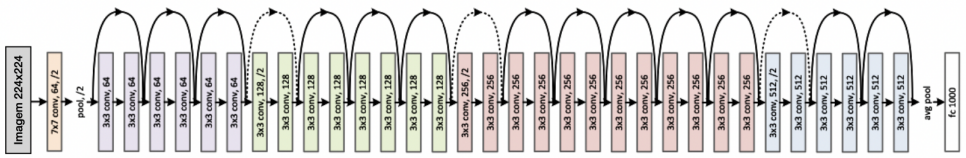

Figura 3.10: ResNet-50 [66].

3.4.2 DINO: Self-Distillation with no Labels

No artigo que introduz o modelo DINO, *Self-**Di**stillation with **no** Labels* [17], os autores propoem a utilização de uma etapa de pré-treinamento auto-supervisionada para modelos ViTs. Dessa forma, o trabalho explora se o

PUC-Rio - Certificação Digital Nº 2112316/CA

sucesso observado com os ViTs está atrelado ao fato de tradicionalmente tais modelos serem treinados de maneira totalmente supervisisonada. A etapa de *self-training* – como os autores chamam o pré-treino – é realizada através de *self-distillation*. Nesse processo, uma rede *student* tenta aproximar as representações que produz com as geradas na saída da rede *teacher* durante o treinamento. As redes apresentam arquiteturas idênticas, porém diferentes hiperparâmetros.

Durante o treinamento, dada uma imagem de entrada x, a rede *student* recebe uma versão transformada aleatoriamente x_1 enquanto a rede *teacher* vê uma diferente, x_2. Os parâmetros da segunda rede são atualizados através de uma média móvel exponencial dos parâmetros da primeira. Ou seja, o gradiente do erro não é retropropagado pelas camadas da rede *teacher*, apenas nas camadas da *student*. As saídas geradas para diferentes *views* de uma mesma imagem de entrada x são alinhadas por meio de uma função de perda de entropia cruzada, que minimiza a distância entre as representações obtidas para cada *view*. Uma vez concluída esta etapa, os modelos podem ser usados em tarefas posteriores, como classificação de imagem e detecção de cópia de imagem. Como no treinamento tradicional de ViTs, pode-se manter a cabeça de classificação ou substituí-la por uma camada densa ao executar uma etapa de *fine tuning* para ajustar o modelo a essas tarefas.

PUC-Rio - Certificação Digital Nº 2112316/CA

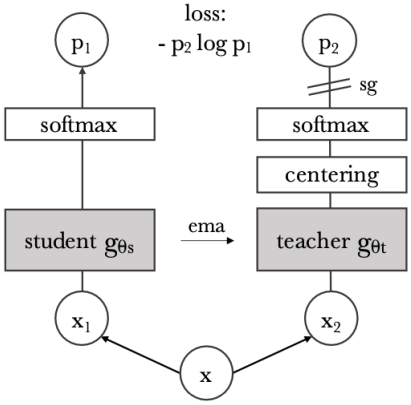

Figura 3.11: DINO

3.4.3 PAWS: Semi-Supervised Learning of Visual Features by Non-Parametrically Predicting View Assignments With Support Samples

No artigo que introdz o método semi-supervisionado PAWS (*Predicting View Assignments With Support Samples*) [16], os autores propõem um procedimento de treinamento que faz um uso eficiente das amostras rotuladas dis-

poníveis. Como na maioria de seus predecessores de aprendizado contrastivo auto-supervisionado [20], o *framework* consiste em passar duas versões transformadas aleatoriamente da mesma imagem para duas redes *encoder* idênticas. No entanto, ao invés de minimizar a função de perda em relação às representações obtidas para cada *view* na camada de saída do *encoder* – como tradicionalmente é feito no aprendizado contrastivo –, o método propõe o uso de um classificador auxiliar baseado em similaridade para gerar *pseudo-rótulos* para cada representação obtida e, em seguida, calcular a entropia com base nesses rótulos.

No PAWS, o modelo gera um vetor de *features*, referente ao vetor de entrada, na saída do *backbone* f_θ. Na sequência, esse vetor passa por uma cabeça de projeção, gerando uma representação. Esse vetor de representações passa por uma cabeça de classificação, que gera o vetor que alimentará o classificador auxiliar. Seja $\hat{x}$ a âncora (uma *view* da imagem) e $\hat{x}^+$ uma amostra positiva (da mesma imagem). A atribuição do pseudo-rótulo é feita comparando as representações obtidas na saída do modelo para as *views* não rotuladas $\hat{x}$ e $\hat{x}^+$ com as de um *batch* amostrado aleatoriamente do conjunto de *rotulado* exemplos de suporte.

A medida de similaridade utilizada no PAWS é a chamada *exponential temperature-scaled cosine*. Essa medida de similaridade é baseada no conceito do produto interno entre dois vetores. No contexto do PAWS, os vetores são as representações obtidas na saída de cada rede. Esse valor do cosseno é escalado a partir de um número inteiro chamado de temperatura, que funciona como um termo regularizador da distribuição de saída aprendida pelo classificador auxiliar. Baseado no cálculo da similaridade entre vetores, o classificador auxiliar, que consiste em um *soft nearest neighbour*, os pseudo-rótulos são atribuídos.

Seja p a predição que o modelo gera para $\hat{x}$, e p^+ a predição para a amostra positiva $\hat{x}^+$. O procedimento de treinamento consiste em otimizar uma função de perda de entropia cruzada $H(p^+, p)$ que tenta maximizar a similaridade entre as atribuições feitas pelo classificador auxiliar para cada vetor obtido para as *views* de uma mesma imagem. O fato dessa otimização ser realizada usando representações e apenas uma partição de amostras rotuladas ajuda a conceber um modelo mais eficiente em rótulos, que também é menos propenso a *over-fitting* [16].

PUC-Rio - Certificação Digital Nº 2112316/CA

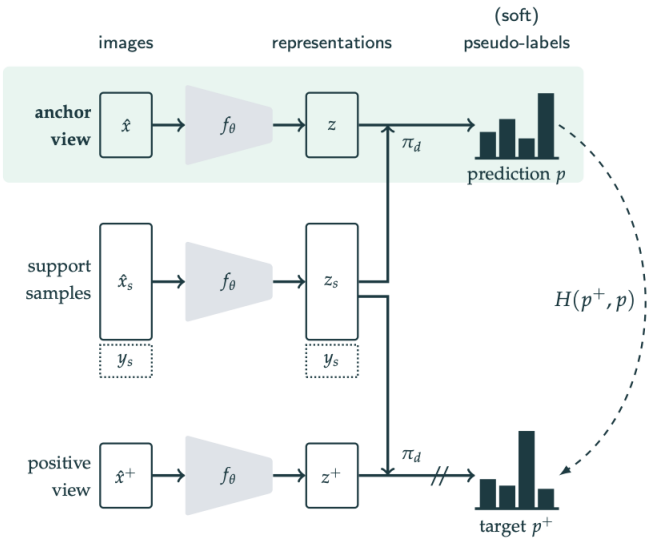

Figura 3.12: PAWS

PUC-Rio - Certificação Digital Nº 2112316/CA

4
Materiais e Métodos

4.1
Conjunto de Dados

Os dados utilizados para todos os experimentos apresentados neste trabalho são imagens amostradas de inspeções submarinas reais. Cada imagem do conjunto de dados é um *frame* cuidadosamente selecionado de um banco de dados de vídeos de inspeções de uma empresa de exploração e produção de petróleo e gás. Como as imagens são privadas, esta seção apresentará uma explicação completa da construção do conjunto de dados e que tipo de objetos aparecem nas imagens.

Os *frames* foram selecionados pelo processo padrão, no qual especialistas assistem atentamente a vídeos elegendo os frames de interesse. Tais *frames* de interesse podem conter qualquer tipo de item ou evento, como por exemplo uma estrutura estática como uma plataforma, um duto, acessórios de dutos, braço robótico do ROV, etc. As classes de interesse do problema compreendem um total de 14 classes, sendo elas: Duto; Enterramento (do Duto ou de algum equipamento); End Fitting; Flange; Sucata; Anodo; Vida Marinha; Resíduo de Anodo; Cruzamento (entre Dutos); Texto (no Duto ou em algum acessório/equipamento); Flutuador; ROV; Corda e Amarra.

Para os experimentos, tem-se um conjunto de dados não-rotulado e um conjunto de dados rotulado. Isso se deve ao fato de que uma das motivações do presente trabalho é validar a mineração do conjunto de dados rotulado, que é consideravelmente menor que a quantidade de *frames* não-rotulados disponíveis na nossa base privada. O conjunto rotulado foi analisado por especialistas em inspeções submarinas, e é formado por um total de 6044 imagens. Já o conjunto não-rotulado é constituído por 30000 imagens, amostradas de uma base com centenas de milhares de *frames* disponíveis. A razão pela qual não foram utilizados todos os *frames* é limitação de processamento computacional e tempo; portanto o presente trabalho é uma aplicação pode ser expandida para o treinamento de novos modelos.

O desequilíbrio de classes é um grande desafio inerente à aplicação. A figura 4.1 apresenta o total de ocorrências para cada classe de interesse no

PUC-Rio - Certificação Digital Nº 2112316/CA

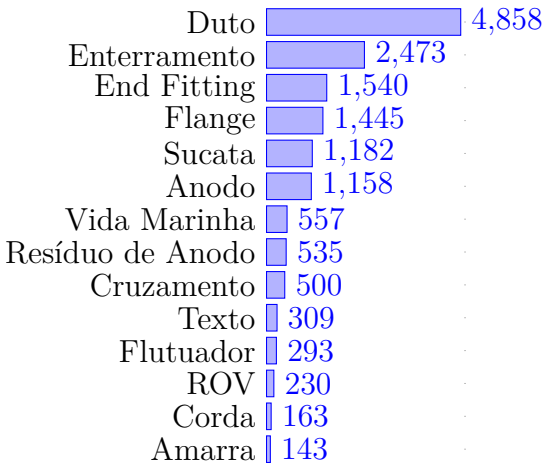

Figura 4.1: Amostras por classe.

conjunto rotulado. Observe que a classe mais frequente é *Duto* com 4858 exemplos enquanto a menos representada é *Amarra* com 143 exemplos. Este é um problema complicado porque a maioria dos *frames* apresenta mais de uma classe e estamos explorando métodos que são semi e auto-supervisionados, portanto as soluções de reamostragem mais comuns [48] não são recomendadas, pois podem alterar a distribuição latente subjacente ao conjunto de dados, e assim afetar negativamente a capacidade do modelo de aprender representações latentes úteis. Além disso, não é incomum que as imagens nas quais as classes menos representadas estejam presentes sejam imagens nas quais dutos e outras classes principais também estejam presentes. Durante o treinamento, o modelo deve ser capaz de aprender a extrair recursos robustos para todas as classes.

Todas as amostras do conjunto de dados são tuplas do formato `[`*image*`,` *labels*`]`. Cada *image* é uma matriz RGB que pode apresentar várias resoluções de pixel e *aspect ratios*. Isso se deve ao fato de que a inspeção submarina pode ser realizada por diferentes empresas especializadas, que usam uma variedade de ROVs/AUVs e uma correspondente diversidade de câmeras. Todas as imagens são remodeladas durante o pré-processamento para um tamanho fixo. Além disso, *labels* é um array codificado em um único formato `[1, c]` onde `c` é o número de classes. Cada *frame* é anotado de acordo com os objetos ou eventos presentes e, portanto, uma imagem pode pertencer a uma ou mais classes.

PUC-Rio - Certificação Digital Nº 2112316/CA

4.2 Método Proposto: Multi-label PAWS (mPAWS)

Esta seção abordará o novo método *multi-label* mPAWS, que é uma adaptação do PAWS para se adequar a um cenário com múltiplos rótulos por imagem. A estrutura original PAWS lida com dados multiclasse. Conforme descrito na seção 3.4.3, a entropia cruzada é otimizada em relação à distância entre as *soft-labels* atribuídas a cada versão transformada da imagem. Para cada época, as *soft-labels* atribuídas correspondem aos rótulos que estão presentes em cada lote de amostras de suporte. Para este cenário, mudar a etapa de pré-treinamento para lidar com dados *multi-label* significaria grandes mudanças no cálculo de perdas e talvez até no classificador auxiliar. Propomos o seguinte framework para mPAWS: (1) pré-processar o dataset de suporte (multi-label), transformando os rótulos de cada imagem em multi-classe; (2) realizar o estágio de pré-treinamento padrão como no PAWS (usando os rótulos multi-classe); (3) executar o estágio de ajuste fino em um cenário *multi-label* (usando os rótulos originais, multi-label). Por fim, para a etapa final do protocolo de avaliação proposto a ser apresentado na seção 4.3.1, utilizamos apenas os modelos ajustados.

O *framework* mPAWS proposto consiste em adicionar ao PAWS uma etapa de pré-processamento inspirada na técnica de *label powerset*. Inicialmente, para o conjunto de dados de suporte, temos uma matriz de rótulos com formato `[num_samples, c]` onde `num_samples` representa o número de amostras do conjunto (660) e `c` o número de classes multi-label (14). Essa matriz é binária, portanto, para uma amostra $[image_i, labels_i]$ do conjunto, o vetor $labels_i$, correspondente à i-ésima linha da matriz de labels, é definido como texttt1 nas posições correspondentes às classes presentes em $image_i$ enquanto o resto de suas posições são definidas como `0`. O objetivo do pré-processamento é transformar cada vetor multi-label $labels_i$ em um único valor positivo, traduzindo o problema para um cenário multi-classe. Para isso, precisamos descobrir quais *combinações únicas de classes* ocorrem no conjunto de dados e substituir esses vetores por valores inteiros positivos, de forma que todas as classes que antes apresentavam combinações de classes multi-label idênticas agora tenham a mesma classe (multi-classe).

A principal diferença de nossa abordagem para o método *labels powerset* tradicional está na combinação de classes [68]. O *label powerset* pega cada combinação única de labels do grupo original de classes multi-label C_k e o traduz diretamente para um identificador para formar um novo conjunto de labels único. Portanto, o procedimento é feito para *todas* as combinações únicas de classes, o que pode levar a um alto desequilíbrio de classes no cenário

PUC-Rio - Certificação Digital Nº 2112316/CA

multi-classe resultante. Para evitar esse problema, nossa abordagem consiste em construir um conversor de classes, que chamamos de *classes encoder*, que atribui o identificador do novo rótulo de maneira um pouco diferente: ele agrupa as combinações menos frequentes em um único *id de classe*, mas primeiro garante que todas as classes multi-label originais já estejam presentes em *pelo menos um* id de classe.

O pré-processamento é feito da seguinte maneira: (1) primeiro, identifica-se quais são todas as *combinações únicas de classes* presentes no conjunto multi-label para formar o vetor $\mathbf{C}$; então, (2) contamos o total de ocorrências para cada uma dessas combinações formando um vetor $\mathbf{O}$. Assim, cada posição i de $\mathbf{O}$ corresponde às ocorrências da combinação de classes a partir da i-ésima posição de $\mathbf{C}$. A partir deles, (3) criamos um *classes encoder* para transformar as classes multi-label em classes multi-classe. Para contornar o problema de combinações de classes com baixo número de ocorrências, o *encoder* combina as combinações menos frequentes em uma única classe no novo cenário multiclasse. Veja o algoritmo 1 para referência.

Algorithm 1 Algoritmo para construção do *Classes Encoder*

Input: vetor de combinação de classes $\mathbf{C}$, vetor de ocorrência de classes $\mathbf{O}$, classes multi-label $\mathbf{C_k}$
Output: multi-class encoder $\mathbf{E}$

threshold final $t_f \leftarrow len(\mathbf{C})$
para cada threshold t_i
for $t_i = 1$ to t_f **do**
  seleciona combinações $\mathbf{C_s} \leftarrow \mathbf{C}[\mathbf{O} \leq t_i]$
  descarta combinações $\mathbf{C_d} \leftarrow \mathbf{C}[\mathbf{O} > t_i]$
  if $\mathbf{C_i} \in \mathbf{C_s}$ for $i = 1, ..., k$ **then**
    atualiza combinações de saída $\mathbf{C_{new}} \leftarrow \mathbf{C_s}$
  else
    break
  end if
end for
for $i = 1$ to $len(\mathbf{C_{new}})$ **do**
  $\mathbf{E}[\mathbf{C_{new}}[i]] \leftarrow i$
end for
$\mathbf{E}[\forall \mathbf{c} \in \mathbf{C_d}] \leftarrow i + 1$
return $\mathbf{E}$

4.3 Protocolo Experimental

Nesta seção, o protocolo experimental é apresentado. Primeiramente, o procedimento utilizado para avaliação dos modelos é introduzido. Em seguida,

PUC-Rio - Certificação Digital Nº 2112316/CA

são explicadas as configurações feitas para cada framework. Finalmente, algumas opções de parâmetros são discutidas.

4.3.1 Protocolo de Avaliação dos Modelos

Uma vez que este trabalho avalia o ganho de desempenho obtido usando menos supervisão para a classificação de imagens submarinas, o protocolo de avaliação segue o utilizado pela maioria dos algoritmos de aprendizado semi-supervisionado e auto-supervisionado. O procedimento consiste em executar uma etapa de pré-treinamento para aprendizado da extração de representações, que pode ser seguida ou não de uma etapa de ajuste fino; e, em seguida, usar o *encoder* ou o modelo do extrator de *features* para ajustar um classificador simples ou realizar *transfer learning* para uma downstream *task* [16, 17]. Por sua simplicidade, o método que escolhemos seguir é o de ajustar um classificador simples.

A sugestão de utilizar um classificador ao invés de uma escolha mais robusta é baseada na motivação do trabalho de avaliar os *frameworks* testados. Empregar um classificador mais complexo nessa etapa desviaria o foco da avaliação da capacidade de aprendizagem de atributos úteis por parte dos *encoders* pré-treinadas. Logo, utilizando um classificador mais elementar, o foco da análise se torna avaliar o modelo gerado pelos *frameworks*, e não o classificador. Ao congelar os pesos do *encoder* e usar as representações obtidas em sua camada de saída como recursos de entrada para um classificador, apenas os pesos do classificador são otimizados.

Portanto, as etapas executadas para todos os experimentos compreenderam o seguinte: (1) uma etapa de pré-treinamento usando a arquitetura e método de escolha (DINO ou mPAWS). A etapa de pré-treinamento pode ser executada do zero – usando pesos inicializados aleatoriamente para as redes –, ou inicializando as redes com os pesos fornecidos pelos autores, correspondentes ao checkpoint final obtido no ImageNet; (1.2) (*somente para mPAWS*) um estágio de ajuste fino supervisionado com as amostras de suporte; (2) treinamento do classificador sobre as representações das redes *encoders* com seus pesos congelados.

4.3.1.1 Pré-treinamento dos Modelos

O experimento padrão é feito com 300 épocas de pré-treinamento ao treinar a rede do zero, e 150 épocas quando inicializamos os pesos com o modelo pré-treinado no ImageNet disponibilizado pelos autores do DINO e

PUC-Rio - Certificação Digital Nº 2112316/CA

do PAWS. Essa diferença do número de épocas se justifica no passo de que incializar os pesos com o ImageNet deve, intuitivamente, ajudar no processo de aprendizagem de atributos que é conferida à etapa de pré-treino.

O pré-treinamento com DINO é feito com uma taxa de aprendizado de $0,0005$ dimensionada para o tamanho do batch, aquecida linearmente durante as primeiras 10 épocas e depois decaída seguindo uma função cosseno. Partindo de uma temperatura de $0,03$, a rede *teacher* tem 50 épocas de aquecimento até o valor de $0,07$. Também usamos o decaimento da taxa de aprendizado por cosseno, de $0,04$ a $0,4$. Os métodos de *augmentation* usados para criar as diferentes versnoes das imagens são: *color jittering*, *gaussian blur* e *multi-crop*. Tais métodos são usados para imitar algumas características de imagens subaquáticas, ou seja, turbidez da água, neve marinha e o movimento do ROV pode produzir visões borradas do mesmo objeto; *multi-crop* imita múltiplas visualizações do mesmo objeto; enquanto o *color jittering* simula diferentes condições de iluminação. Usamos um tamanho de batch de 8 imagens por GPU, o que se traduz em um batch de 32 por usarmos 4 GPUs.

Para mPAWS, durante o estágio de pré-treinamento, usamos uma taxa de aprendizado inicial com valor inicial de $6,4$ e valor final de $0,064$, também decaído via cosseno. Durante as primeiras 10 épocas de aquecimento, a taxa de aprendizagem é aumentada do valor de $0,3$ até $6,4$. Usamos um valor de momento de $0,995$ e para os lotes de suporte, usamos 16 imagens por classe, 3 classes por lote. Usamos um tamanho de lote não supervisionado de 128, o que significa 512 pelas 4 GPUs. A etapa de ajuste fino é feita com uma taxa de aprendizado de $0,02$ e nenhum decaimento de peso por um total de 50 épocas.

As arquiteturas testadas foram as que alcançaram o melhor desempenho (segundo seus autores): ResNet-50 [66] para mPAWS e uma ViT para DINO. O ViT usado para este trabalho é baseado em Transformadores de imagem com eficiência de dados [69], com tamanho de patch de 8. Além disso, realizamos um experimento usando um ResNet-50 para DINO. A fim de comparar e discutir a relevância dos resultados obtidos a partir das estruturas avaliadas, dois experimentos *baseline* totalmente supervisionados foram executados, usando um ResNet-50 e um ViT. Para os modelos supervisionados, os pesos do *encoder* foram inicializados com pesos ImageNet pré-treinados e apenas o classificador foi otimizado.

As mesmas partições de dados do conjunto rotulado foram usadas para todos os experimentos. Os dados foram divididos da seguinte forma: 80% das imagens para o conjunto de treinamento, 10% para validação e os 10% restantes para o conjunto de teste. A partição foi feita de forma estratificada, garantindo que a distribuição das classes-alvo seja aproximadamente a mesma entre as

PUC-Rio - Certificação Digital Nº 2112316/CA

divisões. Isso representa um total de 4.724 amostras para treinamento, 660 amostras para validação e 660 para teste.

Para o DINO, o conjunto não-rotulado e as partições de treino e validação do conjunto rotulado (com seus rótulos descartados) são utilizadas como o conjuntas a partição de treino durante o pré-treinamento. Já para o mPAWS, utilizamos os dados não-rotulados como conjunto principal (apenas a partição de treino), e como conjunto de suporte a partição de treino dos dados rotulados. Para a etapa de ajuste fino desse *framework*, mantemos o conjunto de suporte utilizando-o como dados de treino, e adicionamos a partição de validação como dados para validação.

4.3.1.2 Treinamento do Classificador

O classificador é composto por uma camada densa com uma saída de forma `[1, c]`, onde `c` é o número de classes do nosso problema. Como nossa aplicação consiste em um problema multi-label, usamos uma função de ativação sigmoide na camada de saída. Isso transformará a matriz de *logits* da saída da camada densa em probabilidades para cada classe [70]. O treinamento do classificador é feito da seguinte forma: carregamos e congelamos os pesos do *encoder* obtidos no pré-treinamento. Em seguida, substituímos a cabeça de classificação da rede *encoder* por uma matriz identidade. Por fim, podemos usar o *encoder* como um extrator de *features* para o classificador. O classificador é treinado com uma taxa de aprendizado de $0,0025$ com SGD como otimizador e um tamanho de batch de 32 (8 por GPU) por um total de 150 épocas.

Em cada época de treinamento, um batch de amostras é passado para o *encoder*, que gera representações de tais imagens. Então, essas representações são passadas como entrada para o classificador. A partir da saída do classificador, uma entropia cruzada binária ponderada é calculada, e o gradiente é propagado de volta pela rede. Os pesos usados pela função de custo são as frequências inversas normalizadas de cada classe alvo no conjunto supervisionado. O conjunto de teste é usado apenas para relatar métricas. Portanto, os valores apresentados na seção 5 são os obtidos sobre o conjunto de teste.

Os métodos de *augmentation* usados nos artigos anteriores do DINO e do PAWS para treinar o classificador geralmente consistem em *random multi-crop* e *horizontal flip* durante o treinamento e um *central crop* durante a validação/teste [16, 17]. No entanto, nossos dados são altamente distintos do ImageNet – que é o principal conjunto de dados usado para modelos SOTA – em muitos aspectos: o objeto de interesse não é centralizado; as imagens são coletadas debaixo d'água que apresentam distorção entre outras características

PUC-Rio - Certificação Digital Nº 2112316/CA

complexas; pode haver oclusão de um ou mais objetos em uma imagem, etc. Portanto, usamos apenas *horizontal flip* durante o treinamento e nenhum *augmentation* para validação e teste.

4.3.1.3 Frameworks

Os *frameworks* avaliados são DINO e mPAWS. O *framework* mPAWS é apresentado acima na seção 4.2, onde são introduzidas as modificações propostas para poder aplicar o antigo *framework* a um cenário multi-label. Diferentemente do pré-treinamento semi-supervisionado utilizado pelo mPAWS, o DINO apresenta uma etapa de pré-treinamento auto-supervisionada. Pela forma como este *framework* lida com os dados no pré-treino, sem usar rótulos (hard ou soft), mas apenas representações, nenhuma alteração importante foi necessária para ajustar nossos dados. O objetivo do processo de pré-treinamento é aprender representações úteis para os sinais de entrada por combinando as distribuições de saída das redes de professores e alunos usando uma perda de entropia cruzada.

Os autores apresentam um estudo de segmentação de objetos de vídeo como uma das tarefas a jusante do artigo, que ilustra a capacidade de diferentes chefes dos modelos ViT de atender a diferentes objetos ou regiões semânticas [17]. Os dados utilizados nesse trabalho também compreendem imagens que apresentam vários objetos, então os experimentos foram realizados assumindo que a estrutura pode ser aplicada diretamente ao nosso problema.

PUC-Rio - Certificação Digital Nº 2112316/CA

5
Experimentos e Resultados

Nesse capítulo são apresentados os resultados obtidos para cada configuração testada. Para cada experimento, apresentamos os valores de média e desvio padrão calculados em um total de 30 rodadas. Primeiramente, são apresentados os resultados principais, que correspondem aos experimentos com modelos que utilizaram os hiperparâmetros e partições de dados descritos nas seções acima. Na sequência, são apresentados também resultados adicionais para o DINO e para o mPAWS, na qual algumas hipóteses de melhoria de performance dos modelos foram exploradas.

5.1
Resultados Principais

Para cada configuração, fornecemos três métricas de avaliação: precisão, recall e f1-score. Para cada execução, medimos as métricas seguindo um procedimento baseado em exemplo. Em primeiro lugar, para cada amostra, calculamos cada métrica separadamente com relação às suas próprias previsões; então, após este processo ser feito para todo o conjunto de avaliação, a média é calculada [71]. As métricas obtidas no conjunto de teste estão resumidas na Tabela 5.1. Os modelos supervisionados que foram usados para os experimentos de *baseline* não foram treinados pela autora, os modelos utilizados foram pré-treinados no conjunto ImageNet. Adicionalmente, também informamos o tempo estimado da etapa de pré-treinamento do modelo.

Tabela 5.1: Principais resultados obtidos ao treinar o classificador em cima de *features* extraídas pelos modelos pré-treinados.

Pré-treinamento					Resultado nos Dados de Teste		
Método	Arquitetura	Inic. Pesos	Épocas	Tempo	Precisão	Recall	F1-Score
Supervisionado	RN-50	-	-	-	87.9 ± 0.4	75.3 ± 0.4	77.8 ± 0.2
Supervisionado	ViT-B/8	-	-	-	87.4 ± 0.8	79.3 ± 1.0	$\mathbf{80.1 \pm 0.4}$
DINO	ViT-B/8	Aleatório	300	6d 21h	87.6 ± 0.3	77.1 ± 0.4	79.2 ± 0.3
DINO	ViT-B/8	ImageNet	150	3d 14h	88.8 ± 0.6	82.2 ± 0.7	$\mathbf{82.8 \pm 0.2}$
DINO	RN-50	ImageNet	150	3d 8h	80.9 ± 0.1	61.2 ± 0.1	66.0 ± 0.1
mPAWS	RN-50	Aleatório	300	0d 3h30	84.4 ± 0.8	66.8 ± 1.1	70.6 ± 0.6
mPAWS	RN-50	ImageNet, 1%	150	0d 2h	85.2 ± 0.3	70.8 ± 0.3	73.9 ± 0.2
mPAWS	RN-50	ImageNet, 10%	150	0d 2h	87.6 ± 0.2	71.2 ± 0.2	$\mathbf{75.2 \pm 0.1}$

PUC-Rio - Certificação Digital Nº 2112316/CA

Para os modelos supervisionados utilizados como *baseline*, nota-se que o resultado médio obtido para a ResNet-50 e para o ViT tem uma diferença de 2.3% para o f1-score. A ResNet-50 supervisionada conseguiu alcançar um valor de precisão mais alto e com menor desvio padrão que o ViT, porém o recall obtido para essa rede ficou 4 pontos abaixo. Analisando as equações 3-2 e 3-1, nota-se que isso é um indicativo de que a RN-50 apresenta um número de falsos positivos menor que o ViT, porém um número maior de falsos negativos. Para a aplicação em questão, onde se busca avaliar a viabilidade de embarcar esses modelos em um ROV para inferência em tempo real, a presença de falsos negativos (eventos que existem e que a rede não detectou) é mais crítica que a presença de falsos positivos (eventos que a rede detectou porém não existem). Logo, conclui-se que, em um cenário supervisionado, o ViT apresentou uma capacidade melhor de colaborar com o trabalho dos especialistas.

Para os experimentos com o DINO, o melhor resultado obtido foi para o experimento com o ViT-B/8 inicializado com os pesos do ImageNet fornecido pelos autores. Nota-se uma grande distinção na performance obtida por esse modelo e a RN-50 treinada sob as mesmas condições, com 16.8% de diferença no f1-score médio medido. Logo, assim como avaliado pelos autores, o sucesso do *framework* pode estar atrelado à sua utilização com ViTs e não redes CNNs tradicionais, como é o caso da RN-50. Mais experimentos seriam necessários para validar essa hipótese.

Para os modelos treinados do zero, iniciando os pesos aleatoriamente, os resultados obtidos não foram os mais satisfatórios para o DINO ou para o mPAWS. A RN-50 treinada do zero com o mPAWS atinge um valor de f1-score médio abaixo do obtido pela sua correspondente supervisionada pré-treinada no ImageNet. Já para o DINO, o experimento com ViT-B/8 treinando do zero não ficou abaixo do resultado obtido com a RN-50 para esse mesmo *framework*, e também ficou abaixo do seu correspondente supervisionado. Ambos resultados podem ser justificados pela diferença de tamanho do conjunto de dados utilizado, visto que o ImageNet contém mais de um milhão de imagens enquanto o conjunto de dados de inspeções submarinas utilizados no pré-treino está na casa dos milhares, sendo aproximadamente 40 vezes menor que o primeiro.

Para o mPAWS, examinou-se os resultados obtidos ao inicializar os pesos do *encoder* para o estágio de pré-treinamento com dois modelos diferentes, fornecidos pelos autores. O primeiro foi pré-treinado usando 1% do conjunto de treino do ImageNet como amostras de suporte, enquanto o segundo usa 10% do conjunto. O maior F1-Score médio que obtivemos foi para o experimento inicializado com o modelo treinado com o maior conjunto de suporte, o

PUC-Rio - Certificação Digital Nº 2112316/CA

ImageNet 10%.

O melhor resultado obtido foi com o DINO para o modelo ViT-B/8 inicializado no ImageNet, atingindo 82.8% de f1-score médio, o que corresponde a um ganho ede 2.7% quando comparado com seu correspondete supervisionado. Já o maior f1-score obtido para os modelos mPAWS foi de 75.2%, que pode ser abaixo do melhor resultado do DINO mas é superior em performance ao seu equivalente em arquitetura para esse *framework*. A RN-50 treinada com o DINO obteve uma média fe 66% de f1-score, o que é $9,2\%$ abaixo do resultado obtido pelo melhor modelo mPAWS, e também fica abaixo do seu *baseline* correspondente. Logo, conclui-se que para a RN-50, o *framework* capaz de produzir o melhor resultado dentre os testados é o mPAWS. Isso é mais um indicativo de que um bom resultado com o DINO pode estar subordinado à utilização de ViTs, que é a arquitetura-chefe do artigo que propõe o método.

Na Tabela 5.1, nota-se a discrepância entre o tempo de pré-treinamento do método DINO e do método mPAWS. Comparando uma mesma arquitetura – RN-50 – inicializada com pesos no ImageNet, rodar as 150 épocas de pré-treino do DINO equivale a cerca de 40 vezes o tempo de pré-treino do mPAWS. Nota-se também que o tempo de pré-treino acompanha o número de épocas por experimento, conforme o esperado.

5.2 Resultados Adicionais: DINO

5.2.1 Treinando por mais épocas

Nesse experimento, buscou-se avaliar a influência do número de épocas de pré-treinamento do ViT-B/8 treinado do zero com o *framework* DINO. Para isso, o modelo obtido ao final das 300 épocas foi treinado por mais um adicional de 150 épocas, totalizando 450 épocas de pré-treino. Na Tabela 5.2, nota-se que o f1-score médio obtido para as 30 rodadas apresentou um aumento de 0.1% para o modelo treinado por mais épocas, com uma queda de mesmo valor do desvio padrão. Esse acréscimo ocorre devido à medida de precisão ter abaixado, e o recall aumentado.

O resultado pode ser um indicativo da hipótese de que o modelo pré-treinado por mais épocas manteve estável o número de verdadeiros positivos, porém com um aumento de falso positivos e um decréscimo de falsos negativos. Para a aplicação avaliada, um aumento de falsos positivos não é considerado um cenário crítico pois no contexto de manutenção preventiva, é interessante "pecar por excesso"na detecção de eventos de interesse. Adicionalmente, um

PUC-Rio - Certificação Digital Nº 2112316/CA

Tabela 5.2: Avaliando a influência de aumentar o número de épocas de pré-treinamento para o ViT-B/8 treinado com DINO.

Pré-treinamento					Resultado nos Dados de Teste		
Método	Arquitetura	Inic. Pesos	Épocas	Tempo	Precisão	Recall	F1-Score
DINO	ViT-B/8	Aleatório	300	6d 21h	87.6 ± 0.3	77.1 ± 0.4	79.2 ± 0.3
DINO	ViT-B/8	Aleatório	450	10d 7h	87.0 ± 0.4	77.7 ± 0.4	79.3 ± 0.2

decréscimo de falsos negativos é interessante pois o ROV poderia acelerar com mais certeza.

Outro aspecto importante é o tempo adicional de pré-treinamento do modelo. Como o resultado observado nas métricas é abaixo de 1%, a aplicação deve avaliar se é proveitoso treinar o modelo por mais cerca de 3 dias. Logo, pode-se concluir que aumentar o número de épocas de treinamento para o DINO, quando treinado do zero, é benéfico para a performance do modelo. Porém, em questões práticas, o tempo de pré-treinamento adicional pode ser um fator que contribui com a decisão de não estender essa etapa.

PUC-Rio - Certificação Digital Nº 2112316/CA

5.2.2 Desabilitando o *Multi-crop*

Nesse experimento, buscou-se avaliar o impacto de utilizar o *data augmentation* conhecido como *multi-crop* no pré-treino dos modelos treinados com DINO. Para os experimentos realizados com o ViT-B/8, nota-se que não utilizar esse método de *data augmentation* prejudica o resultado final do classificador, com um decréscimo maior na métrica de recall. Por outro lado, ao remover o *multi-crop* do pré-treino realizado com as redes RN-50, nota-se que a rede apresenta um desempenho muito superior quando não utiliza tal método, com uma diferença de 8.1% no f1-score obtido para as 30 rodadas (Tabela 5.3).

Tabela 5.3: Resultados obtidos sem utilizar *multi-crop* na etapa de pré-treinamento com o DINO.

Pré-treinamento					Resultado nos Dados de Teste		
Arquitetura	Inic. Pesos	Épocas	Multi-crop	Tempo	Precisão	Recall	F1-Score
ViT-B/8	Aleatório	300	✓	6d 21h	87.6 ± 0.3	77.1 ± 0.4	79.2 ± 0.3
ViT-B/8	Aleatório	300		4d 17h	87.2 ± 0.4	75.0 ± 0.5	77.7 ± 0.2
ViT-B/8	ImageNet	150	✓	3d 14h	88.8 ± 0.6	82.2 ± 0.7	82.8 ± 0.2
ViT-B/8	ImageNet	150		2d 11h	88.6 ± 0.7	81.2 ± 0.8	82.2 ± 0.3
RN-50	ImageNet	150	✓	3d 8h	80.9 ± 0.1	61.2 ± 0.1	66.0 ± 0.1
RN-50	ImageNet	150		2d 10h	85.7 ± 0.2	70.1 ± 0.2	74.1 ± 0.1

No artigo que introduz o *multi-crop*, os autores argumentam que a utilização desse método é capaz de aumentar o desempenho do modelo enquanto reduz o viés [72]. Porém, os experimentos realizados pelos autores são em conjuntos de dados nos quais as imagens apresentam certa padronização, sendo as métricas principais obtidas com uma ResNet-50 no conjunto ImageNet. Contudo, os resultados obtidos no problema de classificação de imagens submarinas para a mesma arquitetura indicam limitações nos benefícios que tal método é capaz de conferir, podendo até degradar o resultado. Além dos autores de [72] não realizarem testes em cenários *multi-label*, as classes e imagens de conjuntos de dados *benchmark* apresentam complexidade menor que aplicaçõs do mundo real.

A baixa diferença observada nos resultados do ViT-B/8 se deve ao fato de que esses modelos usam mecanismo de atenção. Por tratarem as imagens em sua entrada como uma sequência de *patches*, é intuitivo que tais modelos possam se beneficiar de métodos de *data augmentation* que se baseiam na extração de *patches* da imagem, como é o caso do *multi-crop*. O mesmo não pode se afirmar para redes CNNs tradicionais, categoria no qual a RN-50 se encaixa. Além do ganho em performance observado para a RN-50 ao realizar o pré-treino sem *multi-crop*, há também um ganho em tempo de otimização de cerca de 1 dia. Esse mesmo ganho em tempo é observado para os modelos treinados com o DINO, porém a queda nas métricas calculadas para estes modelos desencoraja a avaliação desse *trade-off*.

5.2.3 Diminuindo o vetor de saída

Nesse experimento, buscou-se explorar o efeito de diminuir a dimensionalidade do vetor de saída produzido pela rede *encoder*. A Tabela 5.4 apresenta os resultados obtidos treinando o modelo do zero com as seguintes dimensões de saída: 65536 (dimensão originalmente utilizada pelos autores [17]), 32768 (metade da original) e 1024. Para o experimento onde utilizamos um tamanho de vetor correspondente à metade do proposto originalmente pelos autores, nota-se que o resultado obtido é um pouco abaixo do que o obtido com a dimensão original, correspondendo a uma queda de 0.5% no f1-score médio. No entando, para o experimento utilizando o menor valor testado (1024), que corresponde a uma dimesão 64 vezes menor que o vetor original, o resultado obtido é o oposto, apresentando um ganho de performance de 0.5% no f1-score médio.

O resultado obtido pode ser justificado pelo fato de que o conjunto utilizado pelos autores em [17] é o ImageNet, que apresenta milhares de classes.

PUC-Rio - Certificação Digital Nº 2112316/CA

Tabela 5.4: Avaliando a influência do tamanho do vetor de saída para o ViT-B/8 treinado com DINO.

Pré-treinamento					Resultado nos Dados de Teste		
Arquitetura	Inic. Pesos	Dimensão do Vetor	Épocas	Tempo	Precisão	Recall	F1-Score
ViT-B/8	Aleatório	65536	300	6d 21h	87.6 ± 0.3	77.1 ± 0.4	79.2 ± 0.3
ViT-B/8	Aleatório	32768	300	6d 15h	87.4 ± 0.4	76.9 ± 0.5	78.7 ± 0.2
ViT-B/8	Aleatório	1024	300	6d 14h	87.8 ± 0.3	77.6 ± 0.3	79.7 ± 0.2

Já o conjunto de dados do problema de classificação explorado na presente dissertação apresenta apenas 14 classes, então o modelo parece ser capaz de aprender representações mais generalizadas das características das classes do problema utilizando um vetor consideravelmente menor de saída.

O tempo de pré-treinamento dos experimentos apresentou uma estreita queda ao diminuir o tamanho do vetor. Utilizando o vetor de dimensão 32768, há uma redução de 6 horas; enquanto para o vetor de 1024 há uma redução de 7 horas de treino. Realizando uma análise proporcional ao tempo total do experimento mais longo correspondente ao vetor de tamanho original (de dimensão 65536), ambas reduções correspondem a menos de 5% da duração do pré-treino. Logo, o ganho em tempo de otimização ao reduzir o tamanho do vetor de saída é irrelevante.

5.3 Resultados Adicionais: mPAWS

5.3.1 Removendo a cabeça de classificação no pré-treino

No artigo do PAWS [16], os autores argumentam que a cabeça de classificação é incluída na arquitetura da rede apenas para fins de comparação com os demais modelos SOTA e que pode ser benéfico removê-la. A rede *encoder* do mPAWS, assim como a do PAWS, é formada por um *backbone* ResNet-50, uma cabeça de projeção e uma cabeça de classficação. No experimento proposto, remover a cabeça de classificação significa realizar a etapa de pré-treino utilizando na entrada do classificador auxiliar os vetores projetados pela rede *encoder* através da cabeça de projeção, sem passar por uma cabeça de classificação que gere uma predição para essas entradas. Portanto, realizou-se um experimento onde essa cabeça não é incluída no pré-treino, e a rede foi treinada do zero.

Ao contrário do resultado obtido pelos autores, remover a cabeça de classificação ao treinar do zero não apresentou diferença estatística significativa.

PUC-Rio - Certificação Digital Nº 2112316/CA

Tabela 5.5: Avaliando o impacto da utilizacão de uma cabeça de classificação durante a fase de pré-treinamento do mPAWS.

Pré-treinamento				Resultado nos Dados de Teste		
Arquitetura	Inic. Pesos	Cabeça de Classif.	Tempo	Precisão	Recall	F1-Score
RN-50	Aleatório	✓	0d 3h30	84.4 ± 0.4	66.8 ± 1.1	70.6 ± 0.6
RN-50	Aleatório		0d 3h30	84.3 ± 0.5	66.4 ± 0.8	70.6 ± 0.4

Adicionalmente, não houve redução no tempo de pré-treinamento. Nota-se que a precisão cai em 0.1% ao removermos a cabeça, e o recall cai 0.4%. O resultado médio permanece o mesmo por conta da variabilidade dos resultados, visto que no experimento sem a cabeça de classificação, o desvio padrão medido ficou menor para todas as métricas. Isso pode ser um indicativo de que remover a cabeça de classificação durante o pré-treino para o mPAWS pode produzir uma rede *encoder* com maior capacidade de generalização, já que o resultado apresenta menos variabilidade. Contudo, mais experimentos são necessários para avaliar a rede nesse sentido.

PUC-Rio - Certificação Digital Nº 2112316/CA

5.3.2 Concatenando o conjunto de suporte ao conjunto principal

Nesse experimento, foi avaliado o cenário onde o conjunto rotulado é também minerado como parte do conjunto principal, com seus rótulos descartados. Isto é, nesses experimentos, o conjunto principal deixa de ser formado apenas pelo dataset de 30000 images não-rotuladas; são adicionadas a esse conjunto principal as imagens pertencentes ao conjunto de suporte (formado pela partição de treino dos dados rotulados), porém com seus rótulos descartados.

Tabela 5.6: Avaliando o efeito de concatenar o conjunto de suporte ao conjunto principal.

Pré-treinamento			Resultado nos Dados de Teste		
Arquitetura	Inic. Pesos	Dados Concatenados	Precisão	Recall	F1-Score
RN-50	Aleatório		84.4 ± 0.4	66.8 ± 1.1	70.6 ± 0.6
RN-50	Aleatório	✓	84.8 ± 0.7	67.2 ± 0.8	71.5 ± 0.3
RN-50	ImageNet, 1%		85.2 ± 0.3	70.8 ± 0.3	73.9 ± 0.2
RN-50	ImageNet, 1%	✓	88.1 ± 0.2	73.4 ± 0.3	77.0 ± 0.2
RN-50	ImageNet, 10%		87.6 ± 0.2	71.2 ± 0.2	75.2 ± 0.1
RN-50	ImageNet, 10%	✓	88.5 ± 0.3	74.3 ± 0.3	77.7 ± 0.2

Para todos os experimentos testados, realizar essa concatenação do conjunto de suporte ao conjunto principal melhorou o resultado. O maior

salto de performance observado foi para o experimento realizado com a RN-50 inicializada com os pesos do pré-treino com ImageNet 1%, onde a concatenação conferiu um ganho de 3.1% no f1-score médio. Esse resultado positivo pode ser justificado pela presença das imagens de suporte no conjunto principal. Ao realizar a entropia cruzada entre os dados rotulados e não-rotulados, isso provavelmente ajuda a rede no seu processo de aprendizagem.

5.3.3 Alterando o conjunto de suporte

Nesse experimento, adicionou-se a partição de validação dos dados rotulados ao conjunto de suporte utilizado na etapa de pré-treino. Na etapa de *fine-tuning* do mPAWS, as partições foram mantidas conforme os demais experimentos: partição de treino e partição de validação como seus respectivos nomes indicam. Logo, a alteração aqui impactou apenas a etapa de pré-treino, onde os dados de validação antes não eram introduzidos.

Tabela 5.7: Avaliando o efeito de adicionar a partiçnao de validação dos dados rotulados ao conjunto de suporte utilizado no pré-treino do mPAWS.

Pré-treinamento			Resultado nos Dados de Teste		
Arquitetura	Inic. Pesos	Conj. Suporte	Precisão	Recall	F1-Score
RN-50	Aleatório	P. de Treino	84.4 ± 0.4	66.8 ± 1.1	70.6 ± 0.6
RN-50	Aleatório	P. de Treino + Val	83.8 ± 1.0	66.2 ± 1.3	70.1 ± 0.5
RN-50	ImageNet, 1%	P. de Treino	85.2 ± 0.3	70.8 ± 0.3	73.9 ± 0.2
RN-50	ImageNet, 1%	P. de Treino + Val	86.8 ± 0.4	69.6 ± 0.4	73.8 ± 0.2
RN-50	ImageNet, 10%	P. de Treino	87.6 ± 0.2	71.2 ± 0.2	75.2 ± 0.1
RN-50	ImageNet, 10%	P. de Treino + Val	84.8 ± 0.2	69.7 ± 0.3	73.1 ± 0.2

Os resultados obtidos foram inferiores para todos as configurações testadas. Isso pode ser justificado pelo contato com os dados de validação na etapa de pré-treinamento ter gerado um possível sobreajuste da rede *encoder* a esses dados. Contudo, nota-se um pequeno ganho apenas na métrica de precisão do experimento realizado com o modelo inicializado com os pesos no ImageNet 1%. Esse ganho é compensado pela queda de 1.2% no recall, conferindo a rede um resultado inferior ao obtido com a configuração padrão do conjunto de suporte.

PUC-Rio - Certificação Digital Nº 2112316/CA

6
Conclusões e Trabalhos Futuros

Neste trabalho, foi explorado o problema de classificação multi-label de imagens de inspeção submarina. Para tal, foram utilizados dados de imagens extraídos de vídeos de inspeções submarinas reais, pertencentes a uma empresa grande do setor. A aplicação constitui um problema desafiador para ser solucionado a partir de modelos tradicionais supervisionados, visto que dados rotulados são escassos e tais modelos têm performance altamente dependentes de um grande conjunto de dados. Portanto, o foco do presente trabalho foi explorar modelos que sejam menos subordinados à disponibilidade de um grande conjunto de dados rotulado.

Duas abordagens diferentes foram testadas: o *framework* proposto mPAWS, que modifica o método semi-supervisionado PAWS (originalmente multi-classe) de forma a permitir o processamento de dados *multi-label*; e o DINO, um método auto-supervisionado de pré-treino. Os modelos obtidos ao final do pré-treino foram avaliados a partir da utilização destes como extratores de *features* no treinamento de um classificador simples, formado por apenas uma camada densa e uma função sigmoidal. Para comparação, foram realizados experimentos *baseline* com arquiteturas equivalentes às recomendadas pelos autores, treinadas de modo completamente supervisionado no conjunto de dados ImageNet.

Dentre os modelos semi e auto-supervisionado testados com a arquitetura ResNet-50, o resultado mais interessante obtido foi conferido ao mPAWS, inicializado com os pesos do modelo com o pré-treino também no ImageNet, usando apenas 10% dos rótulos do ImageNet como conjunto de suporte, atingindo um f1-score médio de 75.2%. Já resultado obtido para a ResNet-50 com o modelo supervisionado não foi superado pelos *frameworks* testados – um f1-score de 77.8%. Porém, é importante ressaltar que a tarefa supervisionada constitui um desafio muito menor para o modelo. Os modelos supervisionados utilizados como *baseline* foram treinados em todo o conjunto do ImageNet, o qual apresenta mais de um milhão de imagens rotuladas por especialistas, enquanto os modelos propostos, semi e auto-supervisionados, tiveram acesso a no máximo 10% dos rótulos (para o caso do mPAWS, que é semi-supervisionado) ou a nenhum desses rótulos (para o DINO, auto-supervisionado), o que constitui

PUC-Rio - Certificação Digital Nº 2112316/CA

um problema mais laborioso. A ResNet-50 treinada com o DINO apresentou o pior resultado médio para essa arquitetura, o que indica que um bom resultado com esse *framework* pode estar atrelado à arquitetura ViT.

O melhor resultado obtido para a arquitetura ViT-B/8 foi com o DINO, atingindo um f1-score médio de 82.8%, ultrapassando o modelo *baseline* supervisionado em 2.7%. O modelo ViT não foi testado no *framework* mPAWS. Vale ressaltar que os melhores modelos obtidos para o DINO e para mPAWS (o DINO com o ViT e o mPAWS com a ResNet-50) apresentaram diferença em precisão de 1.2%, porém o treinado com o mPAWS apresentou um recall bem inferior.

Os resultados obtidos indicaram a viabilidade do uso de tais modelos para resolver o de classificação de imagens de inspeções submarinas em dutos flexíveis. Os modelos treinados estão a disposição da empresa, podendo ser embarcados no ROV ou na plataforma e, assim, auxiliar os especialistas no processo de identificação de possíveis eventos danosos aos equipamentos submarinos. Um estudo preliminar dos resultados aqui apresentados está sendo enviado para publicação, preservando a privacidade dos dados, de forma que terceiros possam realizar as mesmas adaptações nos *frameworks* testados visando diferentes aplicações do mundo real.

Uma outra contribuição do trabalho diz respeito à pesquisa no cenário da adaptação de métodos multi-classe semi e auto-supervisionados a cenários *multi-label* do mundo real. Isso é altamente relevante pois, em grandes empresas, usualmente tem-se disponível um grande conjunto de dados, porém esses dados não costumam apresentar rótulos em sua totalidade. Logo, os *frameworks* propostos podem ser utilizados em tais cenários, pois demonstraram ser capazes de resolver problemas de classificação auxiliando o trabalho de especialistas.

Como uma proposta para trabalho futuro, pode-se expandir os conjuntos adicionando mais imagens rotuladas e não-rotuladas (que já estão disponíveis em nosso banco de dados privado). Isso pode render melhores resultados para os modelos treinados com o mPAWS proposto, pois ao adicionar mais amostras rotuladas também podemos expandir o conjunto de suporte, o que parece ter um impacto significativo na performance dos modelos. Adicionalmente, aumentar o conjunto de dados parece melhorar muito o desempenho de grandes modelos ViT.

Adicionalmente, para o mPAWS, pode-se ampliar o estudo para testar outras arquiteturas. Os experimentos aqui apresentados limitaram-se à utilização de uma ResNet-50, porém é possível avaliar a performance do *framework* proposto com outras redes convolucionais e também com o ViT. Ainda no

PUC-Rio - Certificação Digital Nº 2112316/CA

contexto de arquiteturas, é possível explorar o efeito de modificações em arquiteturas existentes na literatura para a aplicação em questão.

Já para o DINO, melhorias podem ser implementadas na etapa de pré-treino, principalmente no que diz respeito à *pretext task*. Nessa etapa, o aprendizado de atributos é baseado em métodos de *data augmentation*, com destaque para o *multi-crop*. Logo, é possível explorar uma diversidade de combinações de *data augmentation*, e também propor novos métodos.

PUC-Rio - Certificação Digital Nº 2112316/CA

Referências bibliográficas

[1] BUSBY, R. F.. **Underwater inspection/testing/monitoring of offshore structures**. Ocean Engineering, 6(4):355–491, 1979.

[2] CARVALHO, A. A.; PEREIRA, M. C. B. N. F. J. A. B. B. J. L. F.. **Offshore inspection: perspective and advancement**. Revista Tecnologia, 30(2), Jun. 2010.

[3] JOYE, S. B.. **Deepwater horizon, 5 years on**. Science, 349(6248):592–593, 2015.

[4] ANTONELLI, G.. **Underwater Robots**. Springer Cham, 2013.

[5] FORSYTH, D.; PONCE, J.. **Computer vision: A modern approach.** Prentice hall, 2011.

[6] JIANG, T.; GRADUS, J. L. R. A. J.. **Supervised machine learning: a brief primer**. Behavior Therapy, 51(5):675–687, 2020.

[7] GOODFELLOW, I.; BENGIO, Y. C. A.. **Deep Learning**. MIT Press, 2016. `http://www.deeplearningbook.org`.

[8] BAI, Q.; BAI, Y.. **Subsea pipeline design, analysis, and installation**. Gulf Professional Publishing, 2014.

[9] RAIHAN A, J.; ABAS, P. E. D. S. L. C.. **Review of underwater image restoration algorithms**. IET Image Processing, 13(10):1587–1596, 2019.

[10] JIAN, M.; LIU, X. L. H. L. X. Y. H. D. J.. **Underwater image processing and analysis: A review**. Signal Processing: Image Communication, 91:116088, 2021.

[11] OCEANIC, N.; ADMINISTRATION, A.. **What is marine snow?**, 2020.

[12] WANG, Y.; YU, X. A. D. W. Y.. **Underwater image enhancement and marine snow removal for fishery based on integrated dual-channel neural network**. Computers and Electronics in Agriculture, 186:106182, 2021.

PUC-Rio - Certificação Digital Nº 2112316/CA

[13] DENG, J.; DONG, W. S. R. L. L.; KAI, L.; LI, F.. **Imagenet: A large-scale hierarchical image database**, 2009.

[14] KRIZHEVSKY, A.; HINTON, G.. **Learning multiple layers of features from tiny images**. 2009.

[15] BAI, Q.; BAI, Y.. **Subsea pipeline design, analysis, and installation**. Gulf Professional Publishing, 2014.

[16] ASSRAN, M.; CARON, M. M. I. B. P. J. A. B. N. R. M.. **Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples**. arXiv preprint arXiv:2104.13963, 2021.

[17] CARON, M.; TOUVRON, H. M. I. J. H. M. J. B. P. J. A.. **Emerging properties in self-supervised vision transformers**. In: PROCEEDINGS OF THE IEEE/CVF INTERNATIONAL CONFERENCE ON COMPUTER VISION, p. 9650–9660, 2021.

[18] DOSOVITSKIY, A.; BEYER, L. K. A. W. D. Z. X. U. T. D. M. M. M. H. G. G. S. U. J. H. N.. **An image is worth 16x16 words: Transformers for image recognition at scale**. arXiv preprint arXiv:2010.11929, 2020.

[19] CAI, Z.; RAVICHANDRAN, A. F. P. W. M. M. D. B. R. T. Z. S. S.. **Semi-supervised vision transformers at scale**, 2022.

[20] CHEN, T.; KORNBLITH, S. N. M. H. G.. **Big self-supervised models are strong semi-supervised learners**. CoRR, abs/2006.10029, 2020.

[21] AGÊNCIA NACIONAL DO PETRÓLEO, G. N. E. B.. **Painel dinâmico de produção de petróleo e gás natural**, 2022.

[22] SILVA, E. R.. **Inspeção de dutos submarinos com rov na indústria petrolífera offshore**, 2018.

[23] OF ENERGY, O. W.. **What is a subsea development?**, 2015.

[24] BREIVIK, M.; FOSSEN, T. I.. **Guidance laws for autonomous underwater vehicles**. Underwater vehicles, 4:51–76, 2009.

[25] GAFUROV, S. A.; KLOCHKOV, E. V.. **Autonomous unmanned underwater vehicles development tendencies**. Procedia Engineering, 106:141–148, 2015.

[26] NUNES, P. C.. **Auv para inspeção e mapeamento de dutos e equipamentos submarinos**, 2016.

PUC-Rio - Certificação Digital Nº 2112316/CA

[27] **O uso do rov em atividades de inspeção submarina**, 2020.

[28] **Conheça curiosidades sobre equipamentos de nossos sistemas submarinos**, 2015.

[29] RODRIGUES, J. V. G.. **Análise estrutural de master link em manobras de lançamento de dutos flexíveis**. PhD thesis, Universidade Federal do Rio de Janeiro, 2016.

[30] ALVESA, H.. **Considerações sobre os possíveis mecanismos de corrosão no espaçoo anular de dutos flexíveis**. 2018.

[31] **Subsea corrosion survey**, 2022.

[32] OFFSHORE, P.. **Portfolio: Survey and inspection of pipelines risers and platforms**, 2012.

[33] SUBSEA, B.. **Subsea pipeline, platform structural inspection gal**, 2015.

[34] MARINE, M.. **Tepng - fso unity anchor chain and calm buoy subsea inspection by rov**, 2013.

[35] MARINHO, I. P.. **Lançamento de dutos flexíveis em Águas profundas**, 2014.

[36] TECH SUBSEA INSPECTIONS, V.. **Underwater inspection construction**.

[37] HAARR, M. L.; JANNIKE, F. J. F.. **Global marine litter research 2015–2020: Geographical and methodological trends**. Science of The Total Environment, 820:153162, 2022.

[38] GLOBAL, T.. **Ensuring the structural integrity of deepwater risers**.

[39] **Underwater trash detection**, 2022.

[40] MITCHELL, T. M.. **Machine learning**, volumen 1. McGraw-hill New York, 1997.

[41] RASCHKA, S.. **Model evaluation, model selection, and algorithm selection in machine learning**. arXiv preprint arXiv:1811.12808, 2018.

[42] ZHOU, Z.. **Machine Learning**. Springer Nature, 2021.

[43] BISHOP, C. M.; NASRABADI, N. M.. **Pattern recognition and machine learning**, volumen 4. Springer, 2006.

PUC-Rio - Certificação Digital Nº 2112316/CA

[44] SCHMARJE, L.; SANTAROSSA, M. S. S. M. K. R.. **A survey on semi, self-and unsupervised learning for image classification**. IEEE Access, 9:82146–82168, 2021.

[45] SARKER, I. H.. **Machine learning: Algorithms, real-world applications and research directions**. SN Computer Science, 2(3):1–21, 2021.

[46] LIU, X.; ZHANG, F. H. Z. M. L. W. Z. Z. J. T. J.. **Self-supervised learning: Generative or contrastive**. IEEE Transactions on Knowledge and Data Engineering, 2021.

[47] AZIZI, S.; MUSTAFA, B. R. F. B. Z. F. J. D. J. L. A. K. A. K. S. C. T. N. V. N. M.. **Big self-supervised models advance medical image classification**. In: PROCEEDINGS OF THE IEEE/CVF INTERNATIONAL CONFERENCE ON COMPUTER VISION, p. 3478–3488, 2021.

[48] TAREKEGN, A. N.; GIACOBINI, M. M. K.. **A review of methods for imbalanced multi-label classification**. Pattern Recognition, 118:107965, 2021.

[49] VAN ENGELEN, J. E.; HOOS, H. H.. **A survey on semi-supervised learning**. Machine Learning, 109(2):373–440, 2020.

[50] SOHN, K.; BERTHELOT, D. C. N. C. E. D. Z. Z. Z. H. R. C. A. K. A. L. C. L.. **Fixmatch: Simplifying semi-supervised learning with consistency and confidence**. arXiv preprint arXiv:2001.07685, 2020.

[51] ALY, M.. **Survey on multiclass classification methods**. Neural Netw, 19(1):9, 2005.

[52] DE CARVALHO, A. C. P. L. F.; FREITAS, A. A.. **A tutorial on multi-label classification techniques**. Foundations of computational intelligence volume 5, p. 177–195, 2009.

[53] MADJAROV, G.; KOCEV, D. G. D. D. S.. **An extensive experimental comparison of methods for multi-label learning**. Pattern recognition, 45(9):3084–3104, 2012.

[54] SAMMUT, C.; WEBB, G. I.. **Encyclopedia of Machine Learning**. Springer US, Boston, MA, 2010.

[55] HAYKIN, S.. **Neural networks and learning machines, 3/E**. Pearson Education India, 2009.

PUC-Rio - Certificação Digital Nº 2112316/CA

[56] ROSENBLATT, F.. **The perceptron: a probabilistic model for information storage and organization in the brain.** Psychological review, 65(6):386, 1958.

[57] OH, K.; JUNG, K.. **Gpu implementation of neural networks**. Pattern Recognition, 37(6):1311–1314, 2004.

[58] SCHMIDHUBER, J.. **Deep learning in neural networks: An overview**. Neural networks, 61:85–117, 2015.

[59] LECUN, Y.; BOTTOU, L. B. Y. H. P.. **Gradient-based learning applied to document recognition**. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[60] KRIZHEVSKY, A.; SUTSKEVER, I. H. G. E.. **Imagenet classification with deep convolutional neural networks**. In: Pereira, F.; Burges, C.; Bottou, L. ; Weinberger, K., editors, ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, volumen 25. Curran Associates, Inc., 2012.

[61] RUSSAKOVSKY, O.; DENG, J. S. H. K. J. S. S. M. S. H. Z. K. A. K. A. B. M. B. A. C. F.-F. L.. **ImageNet Large Scale Visual Recognition Challenge**. International Journal of Computer Vision (IJCV), 115(3):211–252, 2015.

[62] SIMONYAN, K.; ZISSERMAN, A.. **Very deep convolutional networks for large-scale image recognition**, 2014.

[63] SZEGEDY, C.; LIU, W. J. Y. S. P. R. S. A. D. E. D. V. V. R. A.. **Going deeper with convolutions**, 2014.

[64] VASWANI, A.; SHAZEER, N. P. N. U. J. J. L. G. A. N. K. Ł. P. I.. **Attention is all you need**. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, p. 5998–6008, 2017.

[65] DEVLIN, J.; CHANG, M. L. K. T. K.. **Bert: Pre-training of deep bidirectional transformers for language understanding**. arXiv preprint arXiv:1810.04805, 2018.

[66] HE, K.; ZHANG, X. R. S. S. J.. **Deep residual learning for image recognition**. In: PROCEEDINGS OF THE IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, p. 770–778, 2016.

PUC-Rio - Certificação Digital Nº 2112316/CA

[67] NAIR, V.; HINTON, G. E.. **Rectified linear units improve restricted boltzmann machines**. In: PROCEEDINGS OF THE 27TH INTERNATIONAL CONFERENCE ON MACHINE LEARNING (ICML-10), p. 807–814, 2010.

[68] SPOLAÔR, N.; CHERMAN, E. A. M. M. C. L. H. D.. **A comparison of multi-label feature selection methods using the problem transformation approach**. Electronic notes in theoretical computer science, 292:135–151, 2013.

[69] TOUVRON, H.; CORD, M. D. M. M. F. S. A. J. H.. **Training data-efficient image transformers & distillation through attention**. CoRR, abs/2012.12877, 2020.

[70] ROBERTS, D. A.; YAIDA, S. H. B.. **The principles of deep learning theory**. arXiv preprint arXiv:2106.10165, 2021.

[71] ZHANG, M.; ZHOU, Z.. **A review on multi-label learning algorithms**. IEEE transactions on knowledge and data engineering, 26(8):1819–1837, 2013.

[72] CARON, M.; MISRA, I. M. J. G. P. B. P. J. A.. **Unsupervised learning of visual features by contrasting cluster assignments**. Advances in neural information processing systems, 33:9912–9924, 2020.

PUC-Rio - Certificação Digital Nº 2112316/CA