UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ESDRAS SOUTO COSTA

Meta Aprendizado para Detecção de Anomalias em Imagens

Recife
2022

ESDRAS SOUTO COSTA

Meta Aprendizado para Detecção de Anomalias em Imagens

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: Inteligência Computacional

Orientador: Prof. Dr. Germano Crispim Vasconcelos

Recife
2022

Catalogação na fonte
Bibliotecária Nataly Soares Leite Moro, CRB4-1722

C837m Costa, Esdras Souto
Meta aprendizado para detecção de anomalias em imagens / Esdras Souto Costa. – 2022.
83 f.: il., fig., tab.

Orientador: Germano Crispim Vasconcelos.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2022.
Inclui referências.

1. Inteligência computacional. 2. Redes neurais. 3. Detecção de anomalias. 4. Meta aprendizado. I. Vasconcelos, Germano Crispim (orientador). II. Título

006.31 CDD (23. ed.) UFPE - CCEN 2022 – 80

ESDRAS SOUTO COSTA

"**Meta Aprendizado para Detecção de Anomalias em Imagens**"

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração: Inteligência Computacional.

Aprovado em: 15/02/2022.

BANCA EXAMINADORA

Prof. Dr. Geber Lisboa Ramalho
Centro de Informática / UFPE

Prof. Dr. Jairson Barbosa Rodrigues
Colegiado de Engenharia da Computação / UNIVASF

Prof. Dr. Germano Crispim Vasconcelos
Centro de Informática / UFPE
(Orientador)

Dedico este trabalho a Deus, ao meu filho Isaías e à minha família que foram a minha força e motivação para a realização deste trabalho.

AGRADECIMENTOS

Primeiramente, agradeço a Deus por tudo, pois nos momentos mais difíceis encontrei conforto e forças em Sua palavra e pela oportunidade de poder realizar uma pós-graduação em uma instituição de ensino como a UFPE em um dos centros mais renomados do Brasil, o CIn. Gostaria também de agradecer a minha família em especial ao meu filho Isaías, de onde encontro motivação, carinho e inspiração, mesmo de pouca idade me falava: "vai dar tudo certo papai" a todo momento e seus abraços recarregadores. À minha mãe e meu pai que sempre acreditaram em mim e seus conselhos que ficarão em mim até o fim de minha vida. Agradeço também a meus colegas de trabalho do CESAR que me apoiaram e ajudaram em vários momentos: Edward Roe e Marcio Dahia. Por fim, agradeço aos professores do CIn em especial meu orientador Germano Vasconcelos, pela oportunidade e ensinamentos transmitidos durante o desenvolvimento desse trabalho.

RESUMO

A detecção de anomalias é uma tarefa relevante em diversos problemas do mundo real, com aplicação em diversas áreas e setores do mercado, como medicina, segurança, finanças, monitoramento, controle de ambientes, inspeção industrial, entre outros. Como exemplos, algumas situações enfrentadas diariamente em que algo pode ser 'anormal' ou fora do comum são acidentes de trânsito, detecção de doenças, fraudes em transações de cartão de crédito e assim por diante. Trata-se de um problema desafiador, uma vez que a definição de 'anomalia' é ambígua, ou seja, qualquer evento que não esteja em conformidade com um padrão de comportamento considerado normal pode ser visto como uma anomalia. Apesar de avanços em trabalhos recentes na detecção automática de anomalias assim como na área de Redes Neurais Profundas, tais modelos ainda demandam uma grande quantidade de dados para que se tire proveito de sua expressividade e desempenho. Sendo assim, o presente trabalho apresenta uma nova abordagem para a detecção semi-supervisionada de anomalias em imagens, quando somente amostras sem anomalias estão disponíveis e são consideradas no treinamento dos modelos. Além disso, considera-se como motivação cenários onde não há abundância de dados e apenas uma quantidade pequena de amostras está disponível para o treinamento. Usando técnicas de meta-aprendizado, em particular MAML, a abordagem proposta é comparada com outros algoritmos estado-da-arte na base de dados MVTec-AD, demonstrando resultados superiores em 29 dos 45 casos avaliados nas tarefas de detecção de anomalias em objetos e texturas nunca vistos em treinamento.

Palavras-chaves: redes neurais; detecção de anomalias; meta aprendizado.

ABSTRACT

Anomaly detection is a crucial task in a variety of real-world problems, with applications in several areas and market sectors, such as health, medicine, security, finance, monitoring, environment control, industrial inspection, among others. For example, some situations faced on a daily basis where something might be considered 'abnormal' or out of the ordinary are traffic accidents, detection of illness, fraud in credit card transactions, and so on. Considered a challenging problem, since the definition of 'anomaly' is ambiguous, in other words, any event that doesn't conform to the behavior considered 'normal' can be seen as an anomaly. Despite the great advances in recent works in anomaly detection as well as in the area of Deep Neural Networks, such models still demand a large amount of data to take advantage of its expressiveness and performance. Therefore, the present work introduce a new approach to the semi-supervised image anomaly detection, when only 'normal' samples are considered in the training of the models. In addition, its considered as motivation scenarios where there is no abundance of data and only a small amount of samples are available in training. Using meta-learning techniques, in particular MAML, the proposed approach is compared with other state-of-the-art algorithms in the MVTec-AD database, showing superior results in 29 of the 45 cases evaluated in the tasks of detecting anomalies in objects and textures never seen in training.

Keywords: neural networks; anomaly detection; meta learning.

LISTA DE FIGURAS

Figura 1 – Exemplo de anomalia pontual 19
Figura 2 – Exemplo de anomalia contextual 20
Figura 3 – Exemplo de anomalia coletiva 21
Figura 4 – O uso de classificação supervisionada para detecção de anomalias . . . 23
Figura 5 – Um arcabouço de uma RNA para o reconhecimento de escrita manual . 24
Figura 6 – Uma operação de convolução com um Kernel **K**, se movendo da esquerda para a direita, é realizada em uma imagem **I** 25
Figura 7 – Demonstra como o córtex visual é organizado 26
Figura 8 – Ilustra como o sistema visual se organiza de forma hierárquica baseado no modelo HMax (RIESENHUBER; POGGIO, 1999) 27
Figura 9 – Demonstra como as primeiras camadas de uma CNN InceptionV1 (SZEGEDY et al., 2015) aprendem as representações de uma imagem 27
Figura 10 – Compartilhamento de parâmetros rígidos para aprendizado multitarefa em RN profundas 29
Figura 11 – Compartilhamento de parâmetros suaves para aprendizado multitarefa em RN profundas 30
Figura 12 – Um exemplo de tarefas de 4-shot 2-class classificação de imagens 31
Figura 13 – Diagrama do MAML 32
Figura 14 – A arquitetura NTM 34
Figura 15 – Arquitetura MANN 35
Figura 16 – A arquitetura da rede neural siamesa convolucional 36
Figura 17 – Estratégia geral do modelo 37
Figura 18 – Demonstra estrutura de um Auto-Codificador 39
Figura 19 – Demonstra estrutura de uma rede GAN 40
Figura 20 – Diagrama do modelo U-Net usado 44
Figura 21 – Diferentes possíveis densidades de máscaras aplicadas na imagem de entrada 44
Figura 22 – Um resumo da abordagem META-RIAD 46
Figura 23 – Imagens de exemplo dos objetos Bottle, Cable, Capsule, Hazelnut e Metal nut da base de dados MVTec-AD. Para cada objeto, é mostrado um exemplo livre de anomalias e um exemplo anômalo. A linha superior mostra toda a imagem. A linha inferior oferece uma visão de perto . . 50
Figura 24 – Imagens de exemplo dos objetos Pill, Screw, Toothbrush, Transistor e Zipper da base de dados MVTec-AD. Para cada objeto, é mostrado um exemplo livre de anomalias e um exemplo anômalo. A linha superior mostra toda a imagem. A linha inferior oferece uma visão de perto . . 51

Figura 25 – Imagens de exemplo das texturas Carpet, Grid, Leather, Tile e Wood da base de dados MVTec-AD. Para cada textura, é mostrado um exemplo livre de anomalias e um exemplo anômalo. A linha superior mostra toda a imagem. A linha inferior oferece uma visão de perto 52
Figura 26 – Diagrama ilustrando a técnica RIAD 55
Figura 27 – Diagrama demonstrando a técnica PaDiM 56
Figura 28 – Diagrama demonstrando a técnica AnoGrad 57
Figura 29 – Resultados em AUC-ROC em relação aos K-shots 64
Figura 30 – Resultados qualitativos da saída do modelo para os objetos com abordagem META-RIAD, capturada após $10-shots$. A coluna da esquerda representa a imagem de entrada, a do meio a imagem produzida pelo modelo e a terceira coluna é o mapa de calor representando a localização da anomalia 66
Figura 31 – Resultados qualitativos da saída do modelo para os objetos com abordagem META-RIAD, capturada após $10-shots$; Imagem apresenta a continuação da Figura 30 67
Figura 32 – Resultados qualitativos da saída do modelo para as texturas com abordagem META-RIAD, capturada após o treinamento da configuração $10-shots$. A coluna da esquerda representa a imagem de entrada, a do meio a imagem produzida pelo modelo e a direita é o mapa de calor representando a localização da anomalia 68
Figura 33 – Resultados qualitativos da saída do modelo para os objetos com RIAD, capturada após $10-shots$. A coluna da esquerda representa a imagem de entrada, a do meio a imagem produzida pelo modelo e a terceira coluna é o mapa de calor representando a localização da anomalia . . . 69
Figura 34 – Resultados qualitativos da saída do modelo para os objetos com RIAD, capturada após $10-shots$; Imagem apresenta a continuação da Figura 33 70
Figura 35 – Resultados qualitativos da saída do modelo para as texturas com RIAD, capturada após o treinamento da configuração $10-shots$. A coluna da esquerda representa a imagem de entrada, a do meio a imagem produzida pelo modelo e a direita é o mapa de calor representando a localização da anomalia. 71
Figura 36 – Amostras das imagens geradas pelo modelo AnoGrad após o treinamento da configuração $10-shots$ em relação a cada objeto. Nas linhas ímpares amostras de imagens de entrada, nas pares a imagem gerada pelo modelo. 72

Figura 37 – Amostras das imagens geradas pelo modelo AnoGrad após o treinamento da configuração $10-shots$ em relação a cada textura. Nas linhas ímpares amostras de imagens de entrada, nas pares a imagem gerada pelo modelo. 73

LISTA DE TABELAS

Tabela 1 – Quantidade de imagens de treinamento e teste por categoria 53
Tabela 2 – Infra-estrutura utilizada para realização dos experimentos 59
Tabela 3 – Resultados dos modelos para *1-shot* na métrica AUC-ROC (objetos) . 60
Tabela 4 – Resultados dos modelos para *1-shot* na métrica AUC-ROC (texturas) . 60
Tabela 5 – Resultados dos modelos para *1-shot* na métrica AUC-ROC (objetos e texturas) 61
Tabela 6 – Resultados dos modelos para *5-shots* na métrica AUC-ROC (objetos) . 61
Tabela 7 – Resultados dos modelos para *5-shots* na métrica AUC-ROC (texturas) 62
Tabela 8 – Resultados dos modelos para *5-shots* na métrica AUC-ROC (objetos e texturas) 62
Tabela 9 – Resultados dos modelos para *10-shots* na métrica AUC-ROC (objetos) 63
Tabela 10 – Resultados dos modelos para *10-shots* na métrica AUC-ROC (texturas) 63
Tabela 11 – Resultados dos modelos para *10-shots* na métrica AUC-ROC (objetos e texturas) 64

LISTA DE ABREVIATURAS E SIGLAS

AUC-ROC	Area Under the Receiver Operating Characteristic Curve
CNN	Convolutional Neural Networks
DL	Deep Learning
GAN	Generative Adversarial Network
GMS	Gradient Magnitude Similarity
LSTM	Long Short-Term Memory
MAML	Model-Agnostic Meta-Learning
MANN	Memory-Augmented Neural Network
ML	Machine Learning
MLPs	Multilayer Perceptrons
MSGMS	Multi Scale Gradient Magnitude Similarity
NTM	Neural Turing Machine
OOD	Out-of-Distribution Detection
PaDiM	Patch Distribution Modeling
ReLU	Rectified Linear Unit
RIAD	Reconstruction-by-Inpainting-based Anomaly Detection
RN	Rede Neural
RNAs	Redes Neurais Artificiais
SVDD	Support Vector Data Description
SVHN	Street View House Numbers
SVM	Suport Vector Machines

SUMÁRIO

1 INTRODUÇÃO 15
1.1 MOTIVAÇÃO 16
1.2 PROBLEMA DE PESQUISA 17
1.3 OBJETIVOS 17
1.4 ORGANIZAÇÃO DO TRABALHO 17
2 FUNDAMENTAÇÃO TEÓRICA 19
2.1 INTRODUÇÃO 19
2.1.1 Anomalia pontual 19
2.1.2 Anomalia contextual 20
2.1.3 Anomalia coletiva 21
2.2 DISPONIBILIDADE DE DADOS 21
2.2.1 Detecção de Anomalias Supervisionada 22
2.2.2 Detecção de Anomalias Semi-Supervisionada 22
2.2.3 Detecção de Anomalias Não-Supervisionada 23
2.3 REDES NEURAIS 24
2.3.1 Redes Neurais Convolucionais 24
2.4 META APRENDIZADO 28
2.4.1 Transferência de Aprendizado 28
2.4.2 Aprendizado Multitarefa 28
2.4.2.1 Compartilhamento de parâmetros rígidos 29
2.4.2.2 Compartilhamento de parâmetros suaves 30
2.4.3 Few-Shot Learning e Meta Learning 30
2.4.4 Abordagens comuns em Meta Aprendizado 31
2.4.4.1 Baseado em Otimização 31
2.4.4.2 Baseado em Modelo 33
2.4.4.3 Aprendizagem por Métrica 35
3 DETECÇÃO DE ANOMALIAS EM IMAGENS 38
3.1 INTRODUÇÃO 38
3.2 ABORDAGENS POR RECONSTRUÇÃO 38
3.3 ABORDAGENS POR MÉTRICA 41
3.4 ABORDAGENS POR META APRENDIZADO 42
4 META-RIAD 43
5 EXPERIMENTOS E ANÁLISE DE RESULTADOS 49
5.1 BASES DE DADOS 49
5.2 DEFINIÇÃO DOS EXPERIMENTOS 54
5.2.1 RIAD: Reconstruction by inpainting for visual anomaly detection . . 54

5.2.2 PaDiM: Patch Distribution Modeling Framework for Anomaly Detection and Localization . . . 55
5.2.3 AnoGrad: Backpropagated Gradient Representations for Anomaly Detection . . . 56
5.2.4 Protocolo de Avaliação . . . 57
5.2.5 Pré-Treinamento . . . 59
5.2.6 Ambiente . . . 59
5.3 RESULTADOS . . . 59
5.3.1 Análise Quantitativa . . . 60
5.3.2 Análise Qualitativa . . . 65
6 CONCLUSÃO . . . 74
6.1 CONSIDERAÇÕES FINAIS . . . 74
6.2 CONTRIBUIÇÕES . . . 74
6.3 LIMITAÇÕES E AMEAÇAS À VALIDADE . . . 75
6.4 TRABALHOS FUTUROS . . . 75
REFERÊNCIAS . . . 77

1 INTRODUÇÃO

A detecção de anomalias se refere ao processo de identificação de algo considerado anormal. Considerado um problema desafiador uma vez que uma anomalia pode ser vista como qualquer evento que não esteja em conformidade com um padrão considerado normal. Também nomeadas como: outliers, observações discordantes, exceções, aberrações, surpresas, novidades, peculiaridades ou contaminantes (CHANDOLA et al., 2009). A detecção de anomalias tem uma ampla variedade de aplicações tais como: identificação de transações fraudulentas em um cartão de crédito, auxilio ao diagnostico de doenças, detecção de intrusão para segurança cibernética, detecção de falhas em sistemas críticos de segurança, identificação de diferentes eventos anormais em vídeos de câmeras de segurança tais como: incêndios, explosões, vandalismo e entre outras aplicações.

Aplicações estas, usam em grande parte algoritmos de aprendizagem, tais como aqueles encontrados em Aprendizado de Máquina ou Machine Learning (ML) e mais recentemente Aprendizado Profundo ou Deep Learning (DL). Apesar de necessitar de uma grande quantidade de dados, tais técnicas baseadas em DL são capazes de tirar proveito dos dados extraindo representações relevantes, demonstrado uma boa capacidade de generalização bem como um crescente desempenho nas mais diversas aplicações (KRIZHEVSKY; SUTSKEVER; HINTON, 2012; SIMONYAN; ZISSERMAN, 2014; RAJPURKAR et al., 2017; ZHANG et al., 2017).

Embora haja avanços significativos na detecção de anomalias, (DEFARD et al., 2020), (KWON et al., 2020), (ZAVRTANIK; KRISTAN; SKOČAJ, 2021), métodos existentes baseados em DL demandam uma quantidade significativa de dados para alcançar um bom desempenho, seja eles rotulados ou não. Além disso, a coleta desses dados para a cobertura de todas as possíveis classes de anomalias pode tornar o processo inviável.

O problema de detecção de anomalias em imagens, de forma geral, tem por objetivo identificar anomalias em instâncias do mesmo objeto ou textura, que é o foco deste trabalho.

Entretanto, a detecção de amostras fora da distribuição, Out-of-Distribution Detection (OOD), também está relacionada com a detecção de anomalias, onde identifica amostras que são consideradas fora da distribuição de treinamento, podendo ser um tipo de imagem diferente daquele tipo de imagem que o modelo foi treinado, além do mais, frequentemente usa-se duas bases de dados como método de avaliação, tais dados de treinamento são nomeados como *in-distribution* e *out-of-distribution*. Trabalhos mais atuais como: (SEHWAG; CHIANG; MITTAL, 2021), (TACK et al., 2020), (MACÊDO et al., 2020), usam bases de dados como CIFAR-10 (KRIZHEVSKY et al., 2009), CIFAR-100 (KRIZHEVSKY et al., 2009), ImageNet (DENG et al., 2009), Street View House Numbers (SVHN) (NETZER et al., 2011), entre outras para avaliar seus objetivos.

É de conhecimento comum que as abordagens baseadas em DL, usualmente apresentam

overfitting quando a quantidade de amostras é muito pequena, bem como no treinamento de novos tipos anomalia não existentes, no qual torna-se difícil obter uma boa generalização em tais cenários (GOODFELLOW; BENGIO; COURVILLE, 2016a). Assim como em pesquisas mais recentes (ZAVRTANIK; KRISTAN; SKOČAJ, 2021), (SCHLEGL et al., 2017), estes também dependem de uma quantidade grande de amostras para alcançar um bom desempenho, assim como no processo de adicionar uma nova classe ou para cobrir novos cenários de anomalias ainda não vistos, neste contexto podendo exigir o retreinamento dos modelos um uma base ainda maior que cubra vários inúmeros casos possíveis.

Para endereçar os problemas supracitados, neste trabalho foi usado meta-aprendizado para tratar tais problemas onde poucas amostras estão disponíveis, mais conhecido como Few-Shot Learning. Em particular no presente trabalho foi usado o algoritmo Model-Agnostic Meta-Learning (MAML) (FINN, 2017) para a realização da detecção de anomalias em imagens, bem como para a adaptação rápida de novos exemplos de classes. Originalmente MAML foi avaliado em tarefas de classificação de imagens, regressão e aprendizagem por reforço, sendo assim foi escolhido pela flexibilidade de adapta-lo a outros tipos de tarefas.

Para a avaliação do método proposto, bem como dos modelos adotados para a comparação, é usada a base de dados MVTec-AD (BERGMANN et al., 2019) em um protocolo de avaliação diferente do naturalmente adotado ao se usar tal base de dados, tal protocolo será descrito na Subseção 5.2.4.

Sendo assim, como pode ser visto no Capitulo 5, o modelo demonstrou resultados promissores em vários cenários avaliados. Além do mais, o algoritmo MAML (FINN, 2017) demonstrou ser capaz de produzir um modelo, no qual, com um número limitado de amostras, obtivesse um bom desempenho. Além disso, tal algoritmo em combinação com um modelo baseado em reconstrução, demonstrou ser capaz de tanto detectar a anomalia em uma determinado objeto ou textura quanto localizar a anomalia na imagem, demonstrando que a abordagem proposta pode aprender com uma quantidade limitada de amostras, detectar e localizar a anomalia contida em uma imagem.

1.1 MOTIVAÇÃO

Abordagens existentes em detecção de anomalias compartilham de um problema comum, quando o modelo aprende a realizar a detecção de anomalias de um determinado conjunto de dados e ao ser testado em outro conjunto nunca visto, sua performance cai (LU et al., 2020).

Para que consiga generalizar bem à cenários distintos, o modelo requer uma boa capacidade expressiva, ou seja, requer que complexidade das funções que podem ser aproximadas pela Rede Neural seja alta, isto pode estar relacionado com a quantidade considerável de camadas de uma Rede Neural profunda (RAGHU et al., 2017).

Além disso, neste contexto pode exigir que o treinamento seja feito em um conjunto de dados que cubra os mais diversos cenários, podendo tornar o processo inviável tanto para aquisição dos dados quanto em uma possível implantação do modelo em dispositivos com capacidades computacionais limitadas.

Em um cenário real de um sistema de detecção de anomalias, naturalmente os exemplos de cenários anormais muitas vezes são raros e somente é disponibilizado uma ou poucas amostras do novo cenário em seu estado normal, o ideal é que o modelo conseguisse especificamente se adaptar a um novo cenário no qual fosse usado mesmo que a quantidade de amostras disponíveis seja limitada.

1.2 PROBLEMA DE PESQUISA

Como treinar um modelo, com poucas amostras, ou seja, com a quantidade de exemplos menor ou igual a dez, para detectar anomalias de um objeto nunca visto em treinamento e somente com amostras consideradas normais?

1.3 OBJETIVOS

Este trabalho tem como objetivo desenvolver uma abordagem que seja capaz de detectar anomalias em imagens e que tenha um bom desempenho, onde somente poucas amostras de um objeto ou textura em particular ditas como sem defeito estão disponíveis.

Para que esse objetivo seja alcançado, serão executados os seguintes objetivos específicos:

1. Desenvolvimento de uma abordagem para a detecção de anomalia em imagens que se adapte a novos tipos de objetos ou texturas com um número restrito de amostras;

2. Avaliação experimental utilizando uma base de dados relevante na detecção de anomalias em imagens, que contenha diferentes tipos de objetos e texturas dentro de um protocolo definido;

1.4 ORGANIZAÇÃO DO TRABALHO

As seções desta dissertação encontram-se estruturadas da seguinte forma:

No capítulo **2 Fundamentação Teórica**, apresenta-se os conceitos e referências usadas no trabalho desenvolvido. No capítulo **3 Detecção de Anomalias em Imagens**, apresenta-se os trabalhos realizados pela comunidade científica, bem como a descrição dos modelos comparados. O método proposto é apresentado no capítulo **4 META-RIAD**. O Capítulo **5 Experimentos e Análise de Resultados**, descreve e discute os resultados

dos experimentos efetuados para análise e avaliação dos métodos desenvolvidos. Por fim, no capítulo **6 Conclusão**, apresenta-se as considerações finais sobre os principais tópicos abordados nesta dissertação e indicações de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

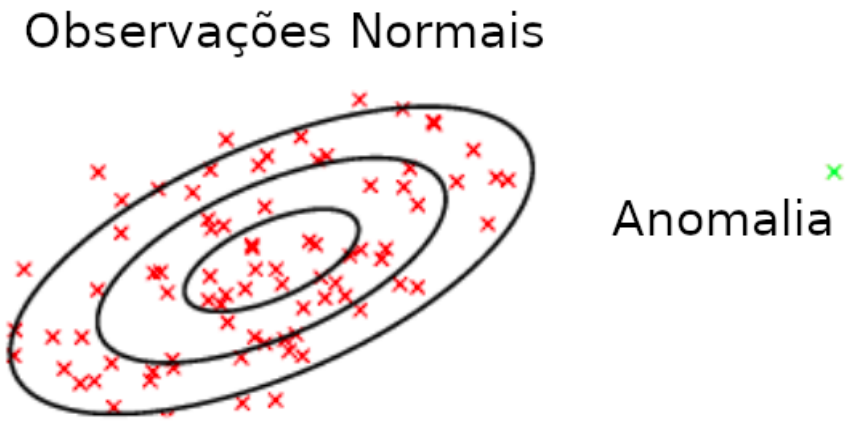
2.1 INTRODUÇÃO

A detecção de anomalias refere-se ao problema de encontrar padrões amostrais não usuais ou anômalos, bem como na identificação de eventos raros ou observações que diferem significativamente da maioria dos dados. Tipicamente amostras não usuais ou itens anômalos são traduzidos para algum tipo de problema seja: uma transação fraudulenta em um cartão de crédito; um defeito no barramento de uma placa eletrônica; um nódulo no tecido mamário em uma imagem mamográfica; entre outros. Segundo (CHANDOLA et al., 2009) Anomalias podem ser divididas em três categorias: Pontuais, Contextuais e Coletivas

2.1.1 Anomalia pontual

Quando uma instância individual ou um ponto em um conjunto de dados é anômalo em relação ao restante dos dados, então pode ser considerada como uma anomalia pontual. Este é o tipo mais simples de anomalia e é o foco da maioria das pesquisas em detecção de anomalia. Um exemplo de aplicação ocorre na detecção falhas em máquinas rotativas, considerando que um exemplo de funcionamento normal um dispositivo funciona com um aceitável de ruido, ao observar que este dispositivo está muito acima da média aceitável de ruido, então pode ser considerado uma anomalia pontual.

Figura 1 – Exemplo de anomalia pontual



Fonte: Adaptado de (ASSYLBEKOV et al., 2016)

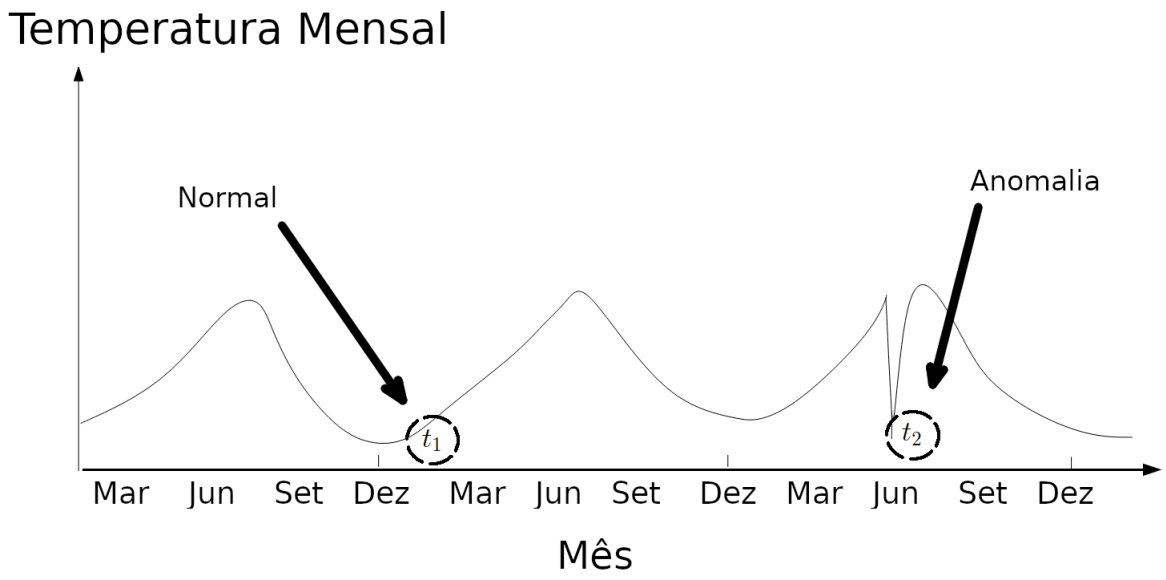
Como podemos ver na Figura 1, uma anomalia pontual pode ser classificada como um ponto em um determinado espaço onde a distância do ponto para os demais diverge

significativamente. Sendo essa significância podendo ser definida por algum limiar, por exemplo, se tal ponto for três vezes mais distante que a média da localização dos demais pontos, esse critério pode ser usado para definir se o ponto pode ser uma anomalia ou não.

2.1.2 Anomalia contextual

Se uma instância de dado é anômala em um contexto específico, mas fora dele não é, então é denominada anomalia contextual. Anomalias contextuais têm sido mais comumente exploradas em dados de séries temporais e dados espaciais. Um exemplo pode ser encontrado em detecção de fraude em cartões de crédito: supondo que, um indivíduo tem um consumo semanal do cartão de R$ 100, exceto durante na semana de natal, quando o consumo alcança R$ 1.000. Uma nova compra de R$ 1.000 é realizada no mês de maio será considerada uma anomalia contextual, uma vez que não está de acordo com o comportamento normal do indivíduo no contexto da época que foi realizada, porém se o mesmo valor fosse gasto durante a semana de natal, o valor gasto seria considerado normal.

Figura 2 – Exemplo de anomalia contextual



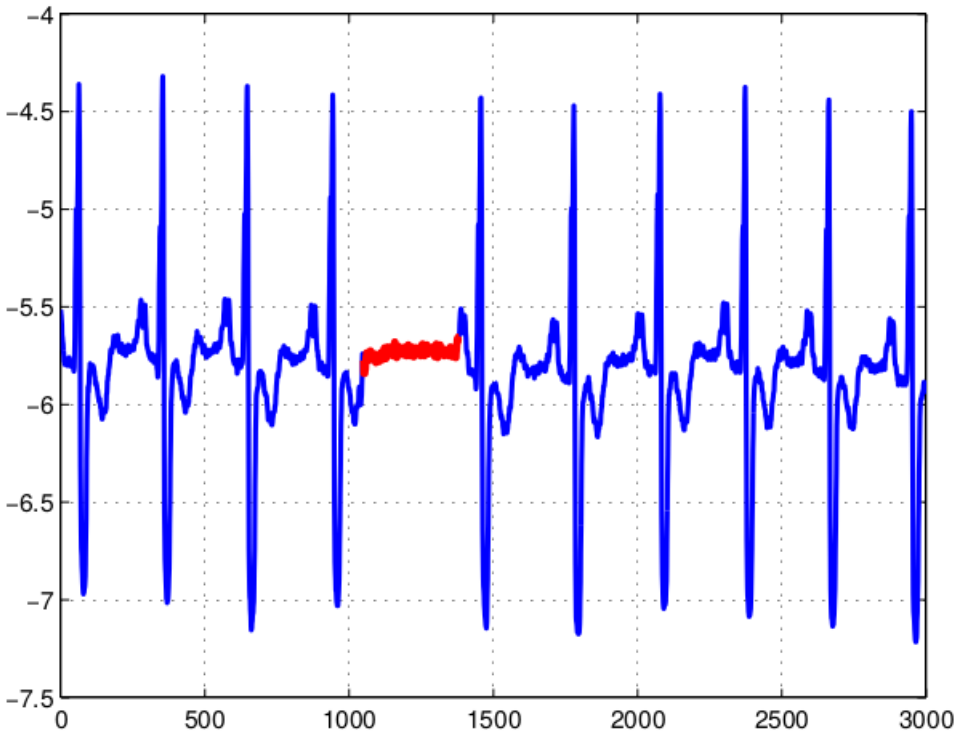
Fonte: Adaptado de (CHANDOLA et al., 2009)

Como pode ser visto no exemplo da Figura 2, em uma anomalia contextual em dados de temperatura média diária durante o ano. No exemplo t_1 e t_2 tem o mesmo valor, porém para o contexto a época do ano onde t_1 foi medida, essa medição pode ser considerada normal, entretanto, vemos que para a época do ano onde t_2 foi medida, essa temperatura é considerada uma anomalia contextual.

2.1.3 Anomalia coletiva

Se uma coleção de instâncias de dados relacionados for anômala com respeito a todo o conjunto de dados, é chamada de anomalia coletiva. Um exemplo pode ser encontrado quando há uma queda repentina nas vendas de uma loja virtual durante algumas horas. É normal ter horas do dia em que as vendas são baixas, como exemplo na madrugada, mas quando vários pontos permanecem coletivamente em baixa, isso é um indicativo de anomalia coletiva.

Figura 3 – Exemplo de anomalia coletiva



Fonte: (CHANDOLA et al., 2009)

Como demonstrado na Figura 3 onde podemos ver um exemplo de um eletrocardiograma humano. A anomalia coletiva, como vista na região destacada, é caracterizada devido ao mesmo valor baixo presente por um tempo anormalmente longo. Devemos notar também que o valor baixo por si só não é uma anomalia.

2.2 DISPONIBILIDADE DE DADOS

Assim como uma tarefa de aprendizado de máquina, a detecção de anomalias se relaciona intimamente com o conjunto de dados a ser analisado, bem como se há rótulos disponíveis para cada classe existente. Os rótulos associados a uma instância de um dado indicam se essa instância é normal ou anômala. A obtenção de dados rotulados que sejam

precisos e representativos é uma tarefa cara. Tal tarefa geralmente é feita manualmente por um especialista humano e, portanto, é necessário um esforço substancial para obter o conjunto de dados de treinamento devidamente rotulado. Além disso, é mais comum que o conjunto de dados não cubram todos os tipos possíveis de comportamento anômalo. Aliás, o comportamento anômalo é frequentemente dinâmico por natureza, por exemplo, novos tipos de anomalias podem surgir, para os quais não há dados de treinamento rotulados. Com base nos rótulos disponíveis, as técnicas de detecção de anomalias podem se dividir em três: Detecção de Anomalias Supervisionada, Semi-Supervisionada e Não Supervisionada

2.2.1 Detecção de Anomalias Supervisionada

Técnicas baseadas em treinamento supervisionado assumem que há disponibilidade dos rótulos no conjunto de dados de treinamento, tanto para as instâncias dos dados normais quanto para as anômalas. Segundo (CHANDOLA et al., 2009), tais técnicas, também conhecidas como detecção de anomalias baseadas em classificação, podem ser sub-divididas em duas amplas categorias: *one-class* e *multi-class*.

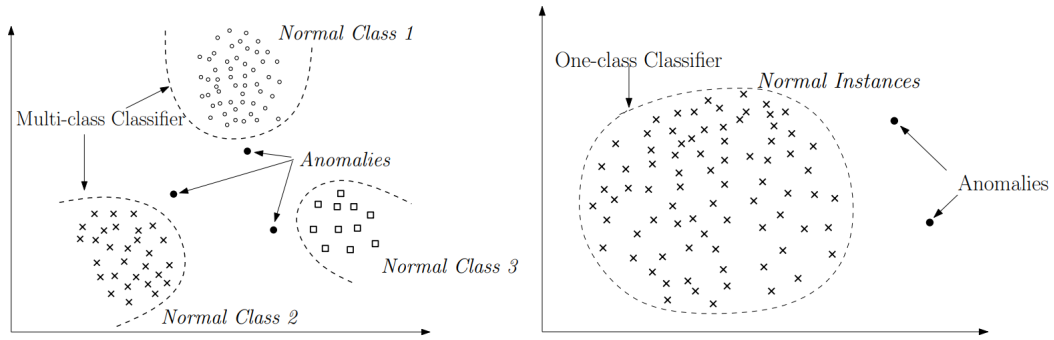
Técnicas de detecção de anomalias baseadas na classificação *multi-class* assumem que os dados contêm instâncias rotuladas pertencentes a várias classes normais. Tais técnicas aprendem a distinguir entre cada classe normal e o restante das classes. Uma instância pode ser considerada anômala se no conjunto de dados de teste ela não for classificada em nenhuma das classes normais. Técnicas de detecção de anomalias baseadas na classificação *one-class* assumem que os dados contêm uma classe, tais técnicas aprendem a distinguir o limite em torno das instâncias normais usando algum algoritmo de classificação. Como exemplo: Máquina de Vetores de Suporte ou Suport Vector Machines (SVM), método proposto por (CORTES; VAPNIK, 1995) no qual constrói um hiperplano de decisão em um espaço de alta dimensão para classificar qual classe uma determina instância de dado pertence, no qual uma instância pode ser classificada como anomalia ou não (SCHÖLKOPF et al., 2001).

2.2.2 Detecção de Anomalias Semi-Supervisionada

Técnicas baseadas em treinamento semi-supervisionado assumem que há escassez ou somente há disponibilidade de uma das classes. Uma abordagem típica usada em tais técnicas é construir um modelo para a classe correspondente ao comportamento normal e usar o modelo treinado para identificar anomalias nos dados de teste no qual podem conter ambas as classes.

Um exemplo de detecção de anomalia semi-supervisionada é a aplicação de Modelos

Figura 4 – O uso de classificação supervisionada para detecção de anomalias



(a) Multi-class Anomaly Detection (b) One-class Anomaly Detection

Fonte: (CHANDOLA et al., 2009)

Gaussianos, onde se assume que os dados são provenientes de uma distribuição Gaussiana, de forma que ao ajustar os parâmetros do modelo, $\bar{x}$ e S, usando estimativas de máxima verossimilhança, obtém assim a pontuação da anomalia (do inglês: Anomaly Score):

$$z = \frac{x - \bar{x}}{S} \tag{2.1}$$

Onde, $\bar{x}$ é a média amostral e S o desvio padrão amostral. É usado então, um conjunto de validação é utilizado para encontrar um limiar ideal acima do qual a instância do dados pode ser considerada uma anomalia.

Tais técnicas semi-supervisionadas tem uma relação comum com métodos estatísticos, como técnicas que usam estimação de funções de densidade de probabilidade e estimativa de densidade por Kernel (TARASSENKO et al., 1995), (DESFORGES; JACOB; COOPER, 1998), (FUJIMAKI; YAIRI; MACHIDA, 2005).

2.2.3 Detecção de Anomalias Não-Supervisionada

Técnicas baseadas em treinamento não supervisionado assumem que não há disponibilidade de rótulos e são baseadas nas propriedades intrínsecas da instância de dados.

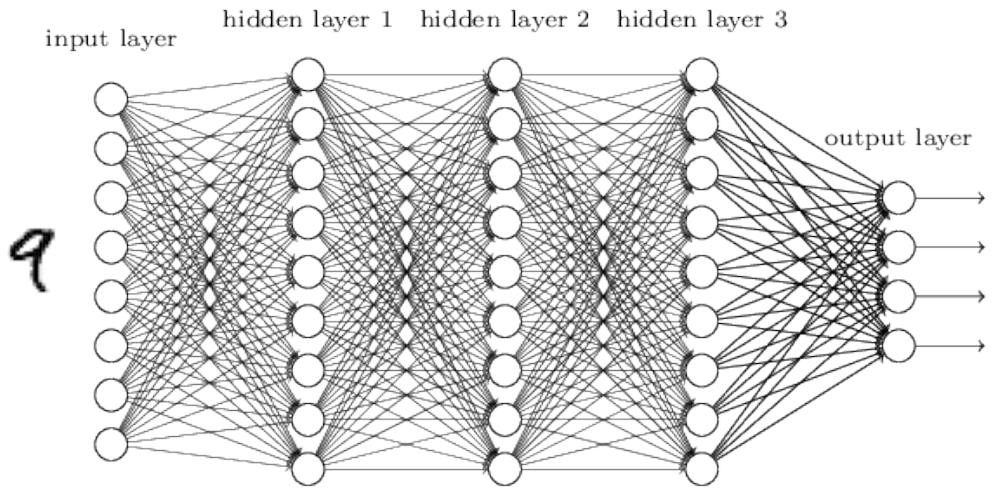
Algumas vezes, a detecção de anomalias semi-supervisionada pode ser vista como não supervisionada, assumindo que um determinado conjunto é conhecido, como o de treinamento.

Tais técnicas são cruciais para o entendimento e a construção de aplicações onde não há rótulos ou são muito escassos. Comumente técnicas são utilizadas técnicas de agrupamento de dados, *clustering*, (YPMA et al., 1997), (YU et al., 2002), (CHAUDHARY et al., 2002), (RAMADAS et al., 2003), assim como métodos baseados em regras de associação, *Association Rules*, (CHEN; LU; TENG, 1990), (MAHONEY; CHAN, 2002), (QIN; HWANG, 2004), entre outros.

2.3 REDES NEURAIS

Inspiradas pelo sistema nervoso central, as Redes Neurais Artificiais (RNAs), são sistemas de neurônios interconectados que podem realizar operações com seus valores de entrada produzindo um estímulo de saída, simulando o comportamento das redes neurais biológicas (MCCULLOCH; PITTS, 1943; ROSENBLATT, 1958; HUSH; HORNE, 1993). Como exemplo no reconhecimento de imagens de escrita manual, pode-se definir um conjunto de neurônios de entrada que quando ativados pelos pixels de uma imagem em sua camada de entrada, produzem um sinal de saída ponderado e então transformado, que é repassado para próximas camadas até o último neurônio que é ativado.

Figura 5 – Um arcabouço de uma RNA para o reconhecimento de escrita manual



Fonte: (NIELSEN, 2015)

Ao adicionar várias camadas escondidas, são conhecidas como Multilayer Perceptrons (MLPs), formalmente seu objetivo é aproximar alguma função f^*. Como exemplo acima de classificação de escrita manual, $\mathbf{y} = f^*(\mathbf{x})$ faz o mapeamento da entrada $\mathbf{x}$ para a categoria $\mathbf{y}$. Sendo esse mapeamento definido como $\mathbf{y} = f(\mathbf{x}; \theta)$ onde a rede neural aprende os valores de θ tal que resulte em uma melhor aproximação da função.

2.3.1 Redes Neurais Convolucionais

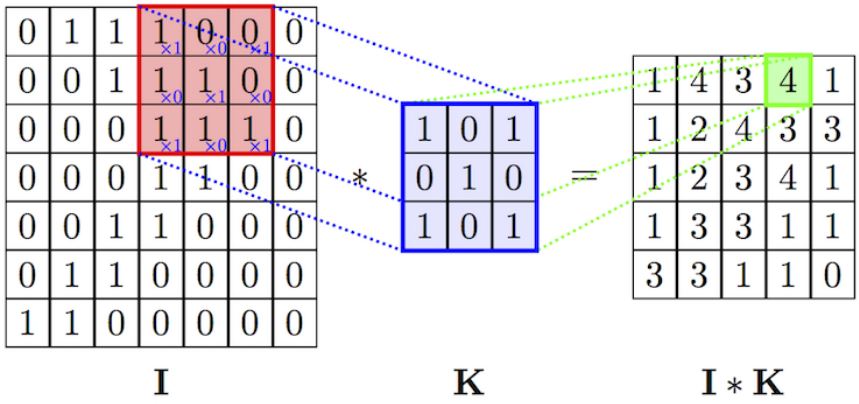
Um tipo especializado uma rede neural são as Redes Neurais Convolucionais, mais conhecidas como Convolutional Neural Networks (CNN) (LECUN et al., 1998). Inspiradas na organização do córtex visual de animais onde neurônios individuais respondem a estímulos apenas em regiões do campo de visão, conhecidas como campos receptivos. Basicamente, cada campo receptivo olha apenas uma pequena região da entrada, esses

campos receptivos podem ser chamados também de Kernels.

O nome “Rede Neural Convolucional” indica que a rede emprega uma operação matemática chamada convolução. Provendo assim características importantes que podem ajudar a melhorar um sistema de aprendizado de máquina dentre elas: interações esparsas, compartilhamento de parâmetros e a capacidade de trabalhar com entradas de tamanho variável (GOODFELLOW; BENGIO; COURVILLE, 2016b).

Por exemplo, ao processar uma imagem, a mesma pode ter milhares de pixels, mas pode-se detectar características pequenas e significativas, como bordas com kernels que ocupam apenas alguns pixels.

Figura 6 – Uma operação de convolução com um Kernel **K**, se movendo da esquerda para a direita, é realizada em uma imagem **I**



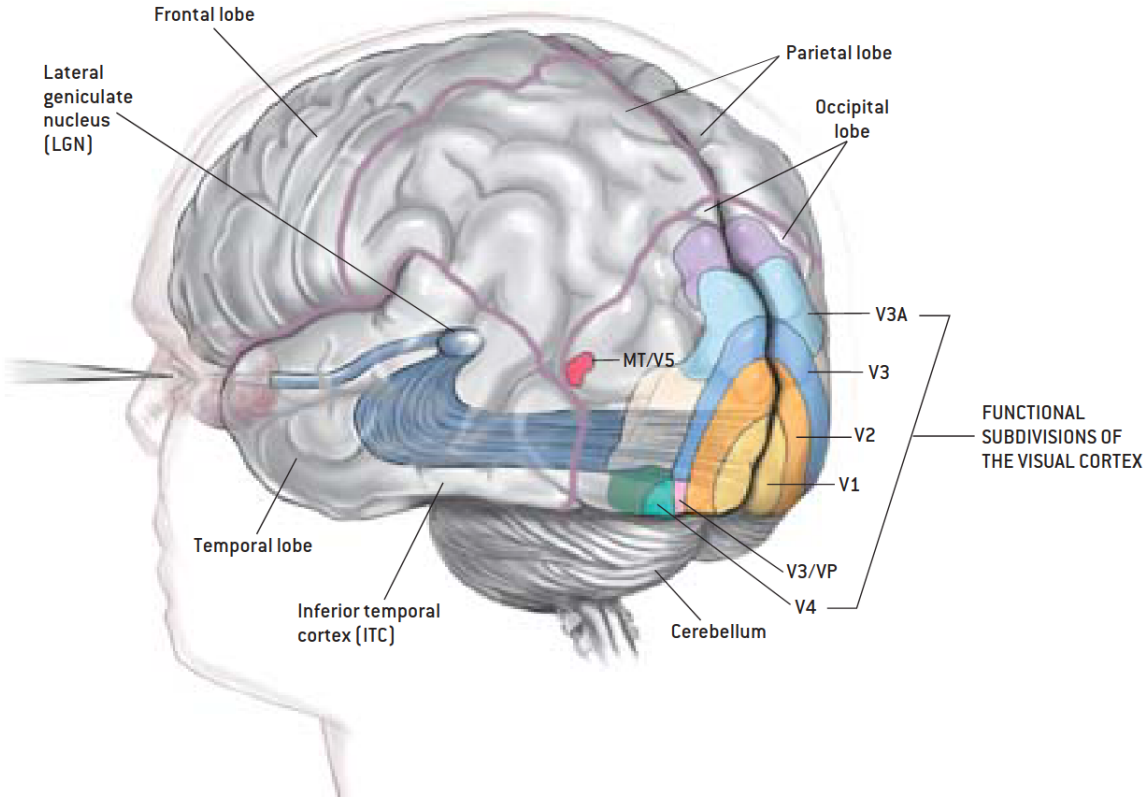
Fonte: O autor (2022)

O Kernel captura características locais da imagem reduzindo a espacialidade a cada operação de convolução, além de que cada kernel compartilha os parâmetros para cada passagem da “janela” do kernel.

Redes Neurais Convolucionais tem tido uma evolução muito rápida dos últimos anos, também foram usadas para ganhar muitas competições. A intensidade do atual interesse comercial na Aprendizagem Profunda começou quando (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) venceu o desafio de reconhecimento de objetos do ImageNet (DENG et al., 2009).

Uma de suas vantagens é sua capacidade de aprender representações hierárquicas, já que também são inspiradas no modelo hierárquico do sistema visual, conforme a rede se torna mais profunda, no sentido de quantidade de camadas empilhadas uma após a outra, a rede aprende representações distintas a cada camada, sendo que as camadas mais próximas da entrada aprendem representações mais básicas como: segmentos de linhas e bordas. As camadas mais profundas aprendem, baseadas nas camadas anteriores, representações mais complexas por exemplo: textura de uma grama, representações de olhos humanos, etc.

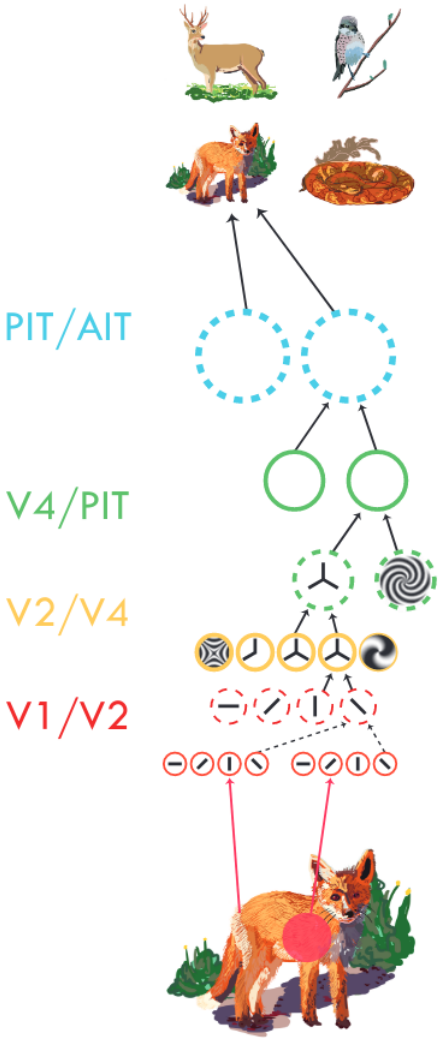
Figura 7 – Demonstra como o córtex visual é organizado



Fonte: (LOGOTHETIS, 1999)

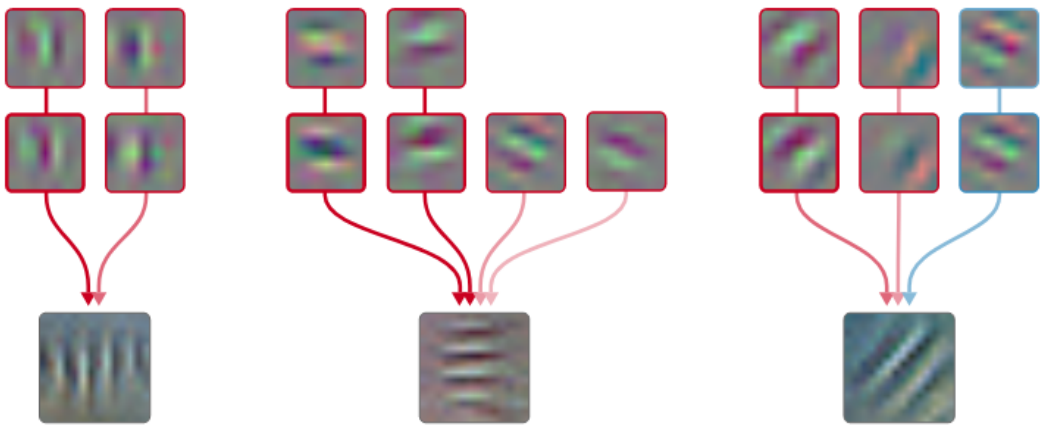
Na Figura 8 vemos uma representação desse modelo, bem como na Figura 9 como essa representação se estende a uma CNN.

Figura 8 – Ilustra como o sistema visual se organiza de forma hierárquica baseado no modelo HMax (RIESENHUBER; POGGIO, 1999)



Fonte: (SERRE, 2013)

Figura 9 – Demonstra como as primeiras camadas de uma CNN InceptionV1 (SZEGEDY et al., 2015) aprendem as representações de uma imagem



Fonte: (OLAH et al., 2020)

2.4 META APRENDIZADO

Os seres humanos tendem a aprender novos conceitos muitas vezes com sucesso a partir de um único exemplo, por outro lado os algoritmos de aprendizado de máquina normalmente exigem dezenas ou centenas de exemplos para executar a mesma tarefa com precisão semelhante. Os paradigmas tradicionais de aprendizado de máquina como o aprendizado supervisionado tem demonstrado grande sucesso nas mais diversas tarefas como classificação, segmentação e entre outras. Entretanto, tais paradigmas exigem abundância de dados para o treinamento de tais métodos, tornando assim a aquisição ou rotulação dos dados uma tarefa custosa e que exige trabalho humano, muitas vezes de especialistas em uma área especifica para exercer tal tarefa.

Meta aprendizado, ou aprender a aprender, pode ser considerado como um conjunto algoritmos capazes treinar modelos de ML que sejam capazes de aprender com uma quantidade pequena de dados, seja de conhecimento anterior ou de outros domínios e se adaptar facilmente a novos ambientes. De forma mais geral, visa a aprender o processo de aprendizado, o tornando mais eficiente de acordo com alguma métrica pré-estabelecida.

2.4.1 Transferência de Aprendizado

A transferência de aprendizado é um método de aprendizado de máquina em que um modelo desenvolvido para uma tarefa é reutilizado como ponto de partida para outro modelo em uma segunda tarefa. A abordagem mais comum é primeiro treinar uma CNN, em uma base de dados como ImageNet (DENG et al., 2009) para uma determinada tarefa e utilizar, agora a CNN pré-treinada, para uma tarefa especifica reaproveitando as características aprendidas no pré-treinamento. Geralmente ao reaproveitar uma CNN pré-treinada, é uma prática comum "congelar" as primeiras camadas dessa CNN, ou seja, tornar os pesos das primeiras camadas imutáveis, não permitindo a atualização desses pesos durante o treinamento para a tarefa específica, pois em sua maioria as características aprendidas nas primeiras camadas de uma CNN são comuns entre diferentes tarefas em Visão Computacional.

2.4.2 Aprendizado Multitarefa

Tipicamente ao treinar um modelo de ML, é desejável que a otimização do modelo seja realizada para minimizar uma métrica em particular, para tanto um modelo único ou um comité de modelos é treinado em uma tarefa desejada. Embora geralmente um desempenho aceitável é alcançado dessa maneira, por estar focado em nossa única tarefa, informações importantes são ignoradas que poderiam ajudar o modelo a alcançar

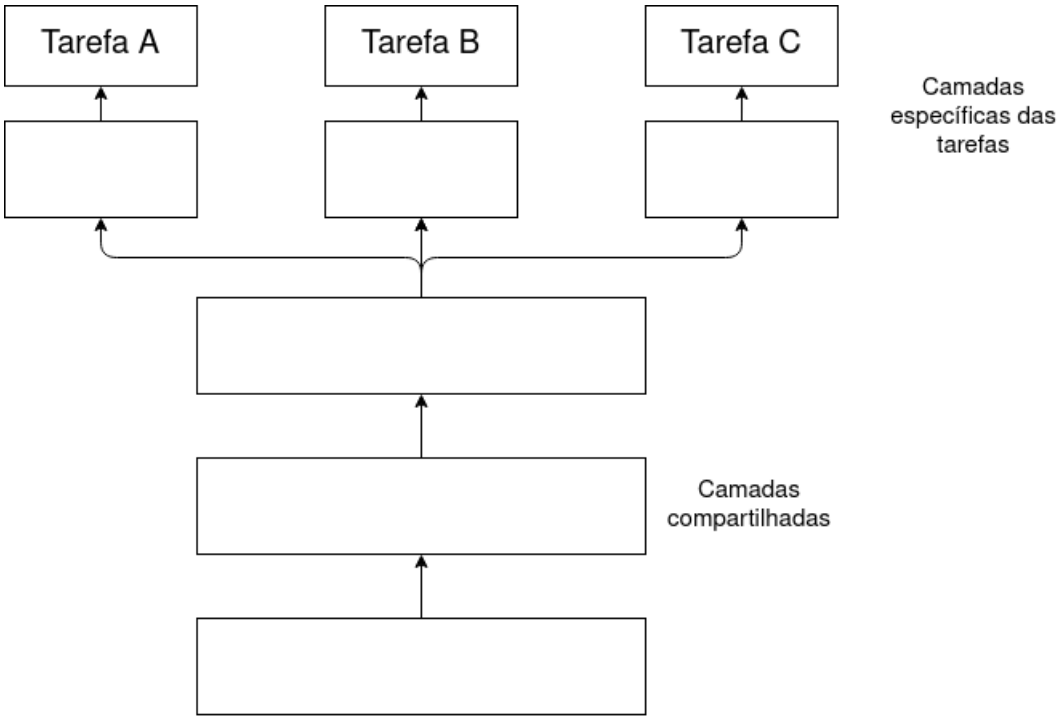
um desempenho ainda melhor na métrica escolhida, essas informações são provenientes de treinamentos de tarefas relacionadas. Compartilhar representações de características aprendidas entre tarefas relacionadas pode permitir que o modelo tenha uma melhor generalização na tarefa desejada (RUDER, 2017).

O aprendizado multitarefa normalmente é feito com o compartilhamento de parâmetros rígidos ou suaves de camadas ocultas de uma Rede Neural (RN) profunda.

2.4.2.1 Compartilhamento de parâmetros rígidos

Geralmente é aplicado compartilhando as camadas ocultas entre todas as tarefas, mantendo várias camadas de saída específicas da tarefa.

Figura 10 – Compartilhamento de parâmetros rígidos para aprendizado multitarefa em RN profundas



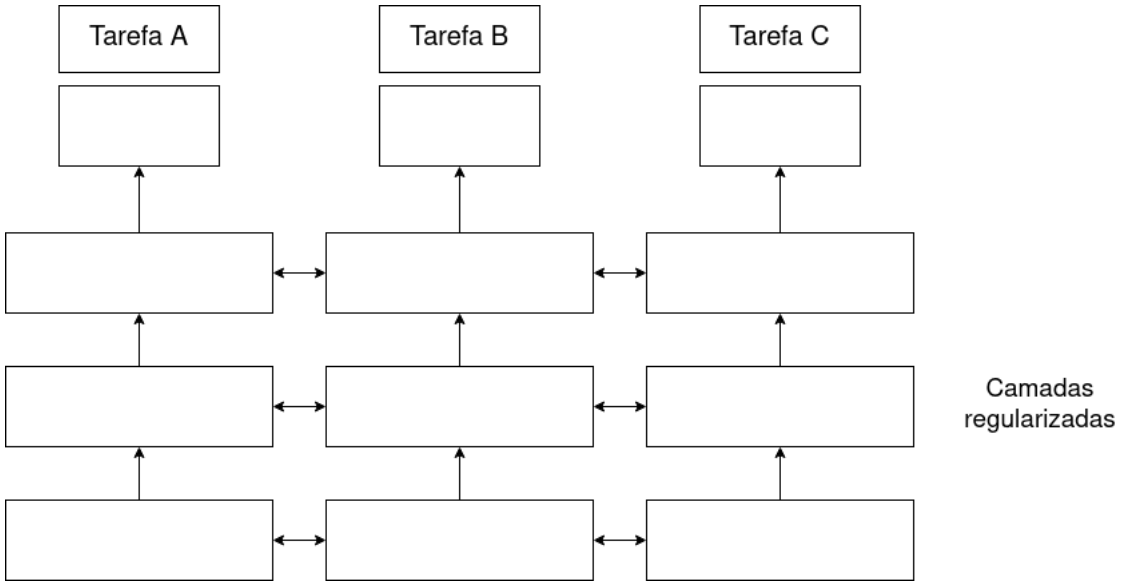
Fonte: O autor (2022)

O compartilhamento de parâmetros rígidos reduz muito o risco de *overfitting*, ou seja, o risco dos parâmetros compartilhados é de ordem N menor que risco dos parâmetros específicos da tarefa, onde N é o número de tarefas. Intuitivamente, quanto mais tarefas são aprendidas simultaneamente, mais o modelo precisa encontrar uma representação que capture todas as tarefas e menor é a chance de *overfitting* nas tarefas.

2.4.2.2 *Compartilhamento de parâmetros suaves*

No compartilhamento de parâmetros suaves, cada tarefa tem seu próprio modelo com seus próprios parâmetros. A distância entre os parâmetros do modelo é então regularizada para encorajar os parâmetros a serem semelhantes, por exemplo, usar a norma $l2$ para regularização.

Figura 11 – Compartilhamento de parâmetros suaves para aprendizado multitarefa em RN profundas



Fonte: O autor (2022)

2.4.3 Few-Shot Learning e Meta Learning

Uma estrutura convencional de treinamento supervisionado para uma tarefa específica é: dado um conjunto de dados de pares de treinamento $\mathcal{D} = \{(x_1, y_1), ..., (x_i, y_i)\}$ onde x_i e y_i são respectivamente os dados de entrada e o rótulo associado à entrada. O modelo é treinado tal que $y_i \approx f_\theta(x_i)$.

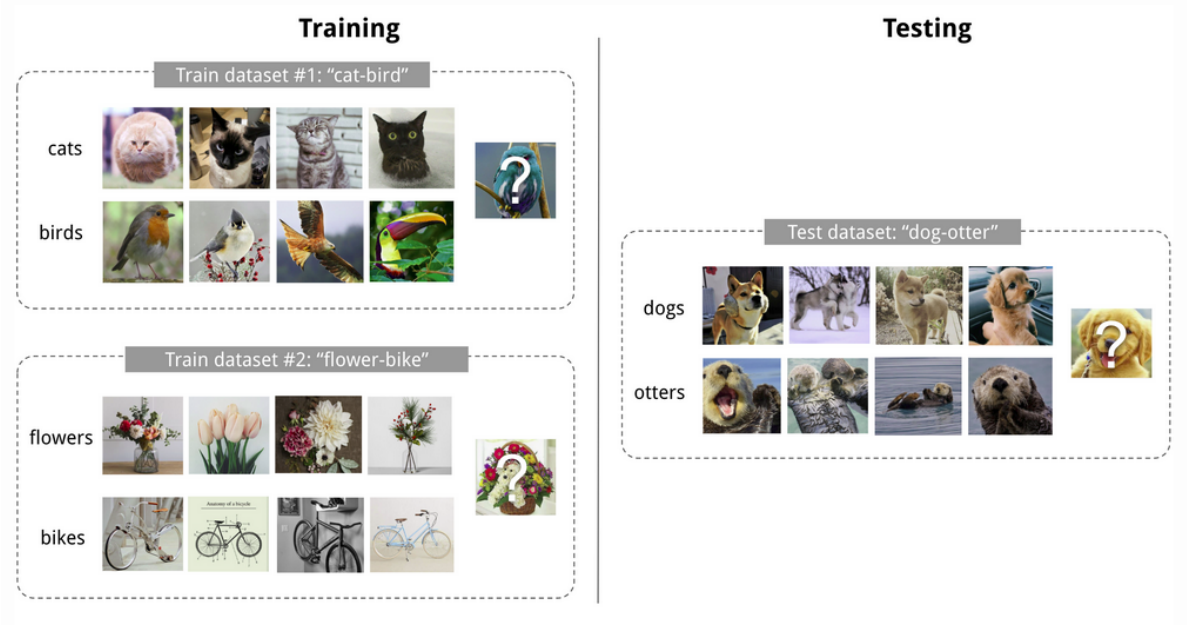
Few-Shot Learing (FSL) é uma instanciação de meta aprendizado, onde o objetivo é realizar com sucesso uma tarefa (seja classificação, segmentação, detecção, entre outras) quando somente alguns exemplos de treinamento estão disponíveis.

Um cenário comum para a estrutura do conjunto de dados $\mathcal{D}$ para o treinamento do modelo baseado em FSL é subdividir o conjunto de dados em duas partes, um subconjunto de suporte (*support set*) $\mathcal{S}$ e um subconjunto de consulta (query set) $\mathcal{Q}$ para validação do modelo. Sendo assim, $\mathcal{D}$ agora é um conjunto de meta-treinamento onde cada linha consiste em uma tarefa $\mathcal{T}_i \sim p(\mathcal{T})$, sendo $p(\mathcal{T})$ a distribuição de probabilidade das tarefas. Sendo assim $\mathcal{D} = \{\mathcal{T}_1, ..., \mathcal{T}_n\}$ onde $\mathcal{T}_i = (\mathcal{D}_i^{\mathcal{S}}, \mathcal{D}_i^{\mathcal{Q}})$ tal que $\mathcal{D}_i^S = \{(x_1, y_1), ..., (x_k, y_k)\}$ e $\mathcal{D}_i^Q = \{(x_1, y_1), ..., (x_l, y_l)\}$. Também é comum a adição de um conjunto de meta-teste $\mathcal{D}^{test}$

com a mesma configuração do conjunto de meta-treinamento, porém suas amostras não estão contidas no conjunto de treinamento, servindo assim para a avaliação da adaptação do modelo a novos tipos de amostras.

A abordagem mais comum é considerar $\mathcal{D}$ um meta-conjunto de dados *N-class K-shot* de tarefas de classificação de dados, sendo que o *support set* comtém K amostras de N classes. Como ilustrado na Figura 12.

Figura 12 – Um exemplo de tarefas de 4-shot 2-class classificação de imagens



Fonte: (WENG, 2018)

2.4.4 Abordagens comuns em Meta Aprendizado

Há três abordagens comuns em Meta Aprendizado que são: Baseado em Otimização, em Modelos e em Métricas.

2.4.4.1 Baseado em Otimização

Abordagens baseadas em otimização tem como objetivo adaptar a otimização de modelos baseados em gradiente de forma que possam lidar com um pequeno número de amostras de treinamento e convergir dentro de um pequeno número de etapas de otimização.

Um exemplo é MAML (FINN, 2017), sendo compatível com qualquer modelo que use o gradiente descendente como otimizador a ser treinado.

De maneira geral, MAML visa aprender uma inicialização do modelo $\theta = \theta_0$, de forma que um pequeno número de etapas internas, ou seja, pequeno número de atualizações nos

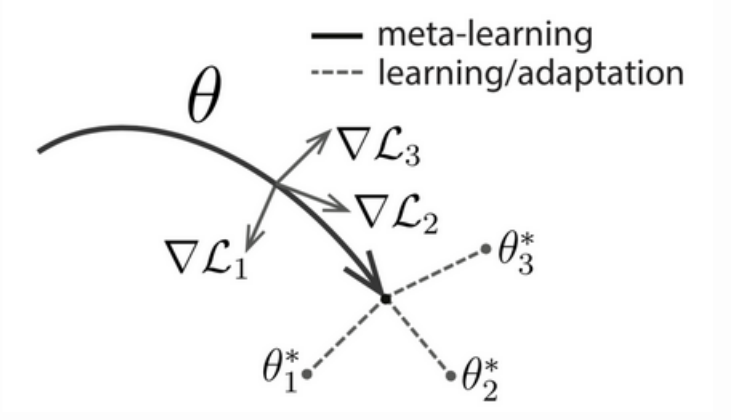
parâmetros via gradiente descendente, produza um classificador com um bom desempenho nos dados de validação. Durante o meta-treinamento, o modelo é treinado para ser capaz de se adaptar para um número determinado de tarefas, podendo ser grande.

Formalmente, com um modelo representado por f_θ com os parâmetros θ, ao adapta-lo para uma nova tarefa $\mathcal{T}_i$, os parâmetros do modelo θ se tornam θ' e então a atualização dos parâmetros via gradiente descendente é realizada em relação a $\mathcal{T}_i$ da seguinte maneira:

$$\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta) \tag{2.2}$$

Onde $\mathcal{L}_{\mathcal{T}_i}$ é a função de perda calculada para a tarefa $\mathcal{T}_i$. O hiper-parâmetro de taxa de aprendizado α pode ser fixo ou também aprendido.

Figura 13 – Diagrama do MAML



Fonte: (FINN, 2017)

Dessa forma para encontrar um θ^* ótimo, é necessário otimizar $f_{\theta_i'}$ sobre o conjunto das tarefas retiradas de $p(\mathcal{T})$ em relação à θ. Sendo assim o meta-objetivo:

$$\min_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'}) = \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)}) \tag{2.3}$$

Observe que a meta-otimização é realizada sobre os parâmetros do modelo θ, enquanto o objetivo é calculado usando os parâmetros do modelo atualizados θ'. Na prática, o método visa otimizar os parâmetros do modelo de modo que um ou um pequeno número de passos no gradiente em uma nova tarefa produza um comportamento que maximize a eficácia nessa tarefa.

A meta otimização sobre as tarefas é realizada via gradiente descendente, tal que a atualização dos parâmetros θ do modelo sejam atualizados da seguinte forma:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'}) \tag{2.4}$$

Sendo β a taxa de meta-aprendizado. O algoritmo completo para o caso geral se encontra descrito no Algoritmo 1.

Algoritmo 1: Model-Agnostic Meta-Learning

Entrada: $p(\mathcal{T})$: distribuição sobre as tarefas
Entrada: α, β: hiper-parâmetros de taxa de aprendizado
1: aleatoriamente inicie θ
2: **enquanto** não convergir **faça**
3: Pegue uma amostra do batch de tarefas $\mathcal{T}_i \sim p(\mathcal{T})$
4: **para todo** $\mathcal{T}_i$ **faça**
5: Avalie $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ em relação a K exemplos
6: Calcule os parâmetros adaptados com gradiente descendente: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **fim do para**
8: Atualize $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **fim do enquanto**

2.4.4.2 Baseado em Modelo

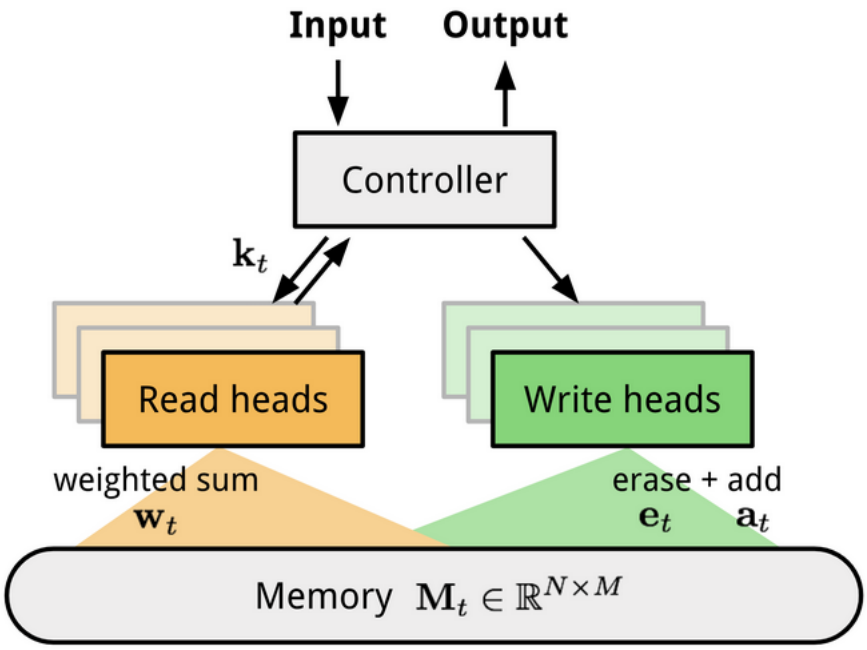
Nesse tipo de abordagem, os modelos são especificamente projetados de maneira tal que a atualização dos parâmetros do modelo proporcione um aprendizado rápido e com pouco treinamento.

Um exemplo é o modelo conhecido como Memory-Augmented Neural Network (MANN) (SANTORO, 2016), sendo classificado como uma família de modelos que usa uma memória externa para auxiliar no processo de aprendizagem. De maneira geral MANN toma como modelo base uma arquitetura Neural Turing Machine (NTM) (GRAVES; WAYNE; DANIHELKA, 2014), na qual acopla uma RN a uma memória externa. Na Figura 14 é demonstrada uma arquitetura NTM.

NTM acopla uma RN denominada controladora (*controller*) a uma memória externa (memory bank), de maneira que o *controller* aprende a ler e escrever na memória externa, enquanto a memória serve de repositório de conhecimento para inferência futura. Os *controllers* podem ser redes recorrentes como Long Short-Term Memory (LSTM) (HOCHREITER; SCHMIDHUBER, 1997) ou uma RN. MANN usa a mesma estrutura de tarefas citadas anteriormente, para tanto é necessário uma forma de codificar e capturar informação para uma nova tarefa que seja estável e facilmente acessada.

O modelo é treinado de maneira que a memória seja forçada a reter as informações por mais tempo até que os rótulos apropriados sejam apresentados mais tarde. Sendo assim, o verdadeiro rótulo y_t é apresentado com um deslocamento a cada interação, ou seja, o modelo vê a sequência de entrada como: $(x_1, null), (x_2, y_1), ..., (x_T, y_{T-1})$. Dessa forma no tempo t o rótulo correto para a amostra anterior (y_{t-1}) é apresentada em conjunto com uma nova amostra x_t, como demonstrada na Figura 16(a). É importante ressaltar que, os rótulos são embaralhados de tarefa-a-tarefa, prevenindo assim que o modelo aprenda lentamente as ligações de rótulo da amostra. Em vez disso, o modelo deve aprender a manter as informações da amostra na memória até que os rótulos apropriados sejam

Figura 14 – A arquitetura NTM



Fonte: (WENG, 2018)

apresentados na próxima etapa de tempo $t+1$, após o qual as informações de rótulo da amostra podem ser associadas e armazenadas para uso posterior.

A memória externa é dada como uma matriz de tamanho $N \times M$, contento N vetores como linhas, cada um tendo M dimensões. O modelo interage com a memória externa por um mecanismo de atenção (soft attention) (XU et al., 2015) através de cabeças de leitura e escrita.

Dado uma entrada $\mathbf{x}_t$, o controller produz um vetor de características $\mathbf{k}_t$, um vetor ponderado de leitura $\mathbf{w}_t^r$ de N elementos é calculado como a similaridade do cosseno entre o vetor de características e cada vetor na matriz de memória $\mathbf{M}_t$, ou seja, cada linha da matriz $\mathbf{M}_t(i)$, sendo assim o vetor de leitura $\mathbf{r}_t$ é a soma ponderada dos registros da memória.

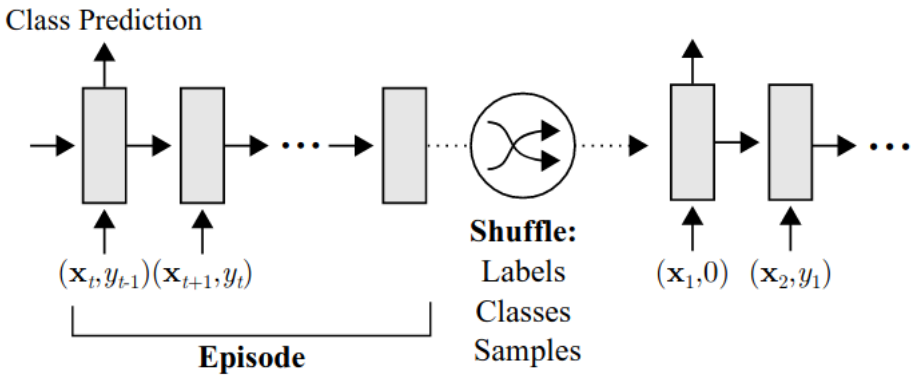
$$\mathbf{r}_t = \sum_i \mathbf{w}_t^r(i)\mathbf{M}_t(i), \text{ onde } \mathbf{w}_t^r(i) = \text{softmax}\left(\frac{\mathbf{k}_t \cdot \mathbf{M}_t(i)}{\|\mathbf{k}_t\|\|\mathbf{M}_t(i)\|}\right) \tag{2.5}$$

Basicamente ao escrever na matriz de memória no tempo t, a cabeça de escrita primeiro remove o conteúdo antigo de acordo com um vetor $\mathbf{e}_t$ e então adiciona a informação nova de acordo com o vetor $\mathbf{a}_t$.

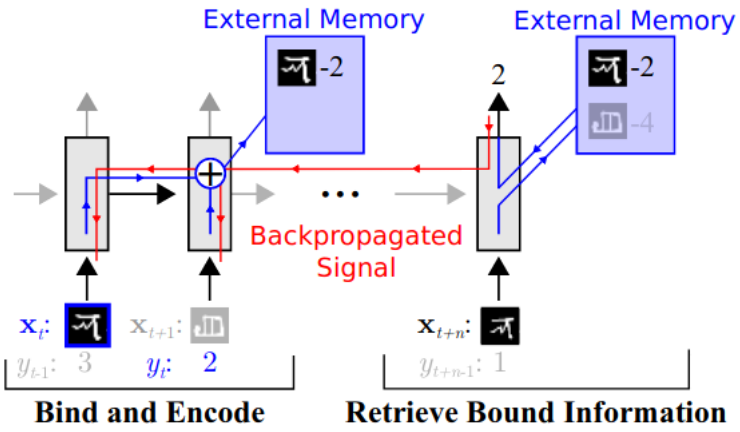
$$\tilde{\mathbf{M}}_t(i) = \mathbf{M}_{t-1}(i)[\mathbf{1} - w_t(i)\mathbf{e}_t]; \text{ remove conteúdo antigo} \tag{2.6}$$

$$\mathbf{M}_t(i) = \tilde{\mathbf{M}}_t(i) + w_t(i)\mathbf{a}_t; \text{ adiciona conteúdo novo} \tag{2.7}$$

Figura 15 – Arquitetura MANN



(a) Configuração das Tarefas



(b) Estratégia do Modelo

Fonte: (SANTORO, 2016)

2.4.4.3 Aprendizagem por Métrica

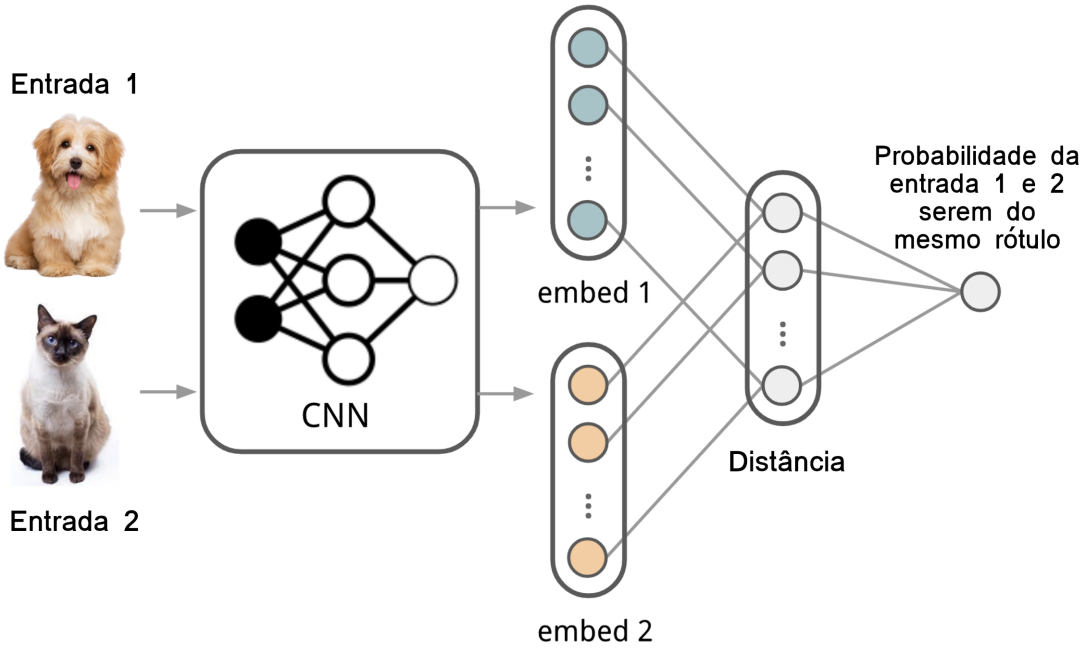
A ideia geral dessa abordagem é aprender uma métrica de distância para mensurar o quão próximas duas instâncias de dados estão, baseados em uma representação gerada por uma vetor de características denominado como *embedding*.

O conceito de métrica de distância pode ser particular do problema, podendo também usar conhecimento *a priori* do domínio, porém muitas vezes se torna difícil projetar métricas que sejam adequadas aos dados e tarefas específicas de interesse.

Sendo assim, se tratando de aprender boas representações de características de maneira que seja computacionalmente viável, dado a alta dimensionalidade dos dados como imagem e, para alguns casos, seja capaz de aprender tais representações, onde com apenas poucas amostras de uma tarefa específica seja capaz de produzir bons resultados, Redes Siamesas (Siamese Neural Networks) (BROMLEY et al., 1993) se dispõe a resolver tais problemas.

Esse tipo de modelo é composto por duas RNs, comumente denominadas de redes siamesas, pois compartilham os mesmos pesos e parâmetros, são treinadas sobre uma função onde tem como objetivo aprender a relação entre pares de amostras de dados de entrada.

Figura 16 – A arquitetura da rede neural siamesa convolucional



Fonte: O autor (2022)

O trabalho descrito em (KOCH et al., 2015), o autor estende a ideia de redes siamesas ao incluir uma rede CNN para aprender uma função que codifica os dados de entrada em um vetor de *embeddings*, como demonstrado na Figura 16.

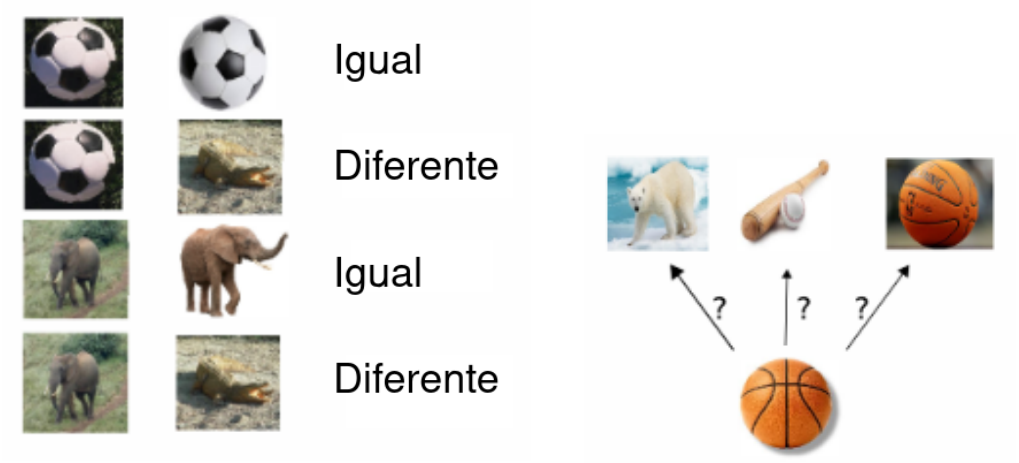
O processo consiste em dois passos:

1. O modelo é treinado para discriminar entre uma coleção de pares iguais/diferentes, Figura 18(a).

2. O modelo e avaliado em novas categorias com base no mapeamentos no vetor de características aprendidas com poucas amostras, Figura 18(b).

Mais precisamente, a CNN aprende a codificar duas imagens por uma função de embedding f_θ e calcula-se a distância L1 entre os dois vetores gerados $|f_\theta(\mathbf{x}_i) - f_\theta(\mathbf{x}_j)|$, seguida de uma camada totalmente conectada e a aplicação de uma função de ativação sigmoid σ, representando a probabilidade de duas imagens serem tiradas da mesma classe.

Figura 17 – Estratégia geral do modelo



(a) Tarefas de verificação (b) Tarefas de One-Shot

Fonte: (KOCH et al., 2015)

Dessa forma, a função de perda é uma função de entropia cruzada, devido ao rótulo ser binário:

$$p(\mathbf{x}_i, \mathbf{x}_j) = \sigma(\mathbf{W}|f_\theta(\mathbf{x}_i) - f_\theta(\mathbf{x}_j)|) \tag{2.8}$$

$$\mathcal{L}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{(\mathbf{x}_i, \mathbf{x}_j, y_i, y_j) \in B} \mathbf{1}_{y_i = y_j} \log p(\mathbf{x}_i, \mathbf{x}_j) + (1 - \mathbf{1}_{y_i = y_j}) \log(1 - p(\mathbf{x}_i, \mathbf{x}_j)) \tag{2.9}$$

Após o modelo ser treinado, em tempo de inferência é fornecido um conjunto de suporte $\mathcal{S}$ e uma imagem de teste $\mathbf{x}$, a predição do rótulo da imagem então é calculada:

$$\hat{c}_S(\mathbf{x}) = c(\arg\max_{\mathbf{x}_i \in S} P(\mathbf{x}, \mathbf{x}_i)) \tag{2.10}$$

onde $c(\mathbf{x})$ é o rótulo da imagem $\mathbf{x}$ e $\hat{c}(.)$ é o rótulo previsto.

A suposição é que o vetor de *embedding* aprendido pode ser generalizado de modo a ser útil em medir a distância entre imagens de categorias desconhecidas.

3 DETECÇÃO DE ANOMALIAS EM IMAGENS

3.1 INTRODUÇÃO

A detecção de anomalias em imagens é um problema importante e desafiador. Com a suposição de ambiente aberto, ou seja, que operam com dados do mundo real sem depender de rótulos ou dados estruturados, espera-se que sistemas de aprendizado desenvolvidos para detecção de anomalias aproveitem o percepção do conhecido (dados e padrões normais) para inferir as incógnitas (padrões anormais ou novos que diferem dos normais).

Em diferentes granularidades de detecção visual, a detecção de anomalias visuais pode ser agrupada em duas categorias: detecção de anomalias visuais em nível de imagem e em nível de pixel. A detecção em nível de imagem geralmente se concentra apenas na questão de saber se toda a imagem é normal ou anormal, enquanto a detecção de anomalia em nível de pixel requer ainda detectar ou localizar as regiões anormais na imagem.

Também conhecida como detecção de anomalia visual, tem sido frequentemente objeto de temas de pesquisas e avanços tecnológicos em ML e DL, bem como ampla literatura já desenvolvida (CHANDOLA et al., 2009), (CHALAPATHY; CHAWLA, 2019), (PANG et al., 2021). Esta seção fornecerá um resumo sobre as abordagens mais recentes.

3.2 ABORDAGENS POR RECONSTRUÇÃO

Modelos baseados em reconstrução tentam modelar o problema com a premissa de que as amostras de treinamento são livres de anomalias, desta maneira podendo ser considerado um problema semi-supervisionado, portanto durante o treinamento o modelo aprende a reconstruir a imagem baseando-se neste conhecimento que é aprendido durante o processo de aprendizagem da rede neural.

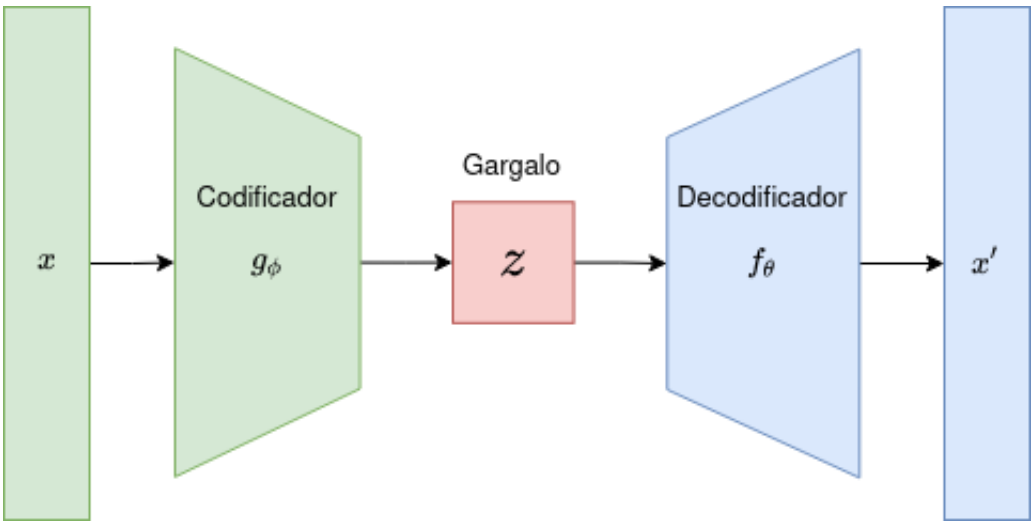
Sendo assim, dado uma imagem de teste com anomalia, tais modelos não devem ser capazes de reconstruir adequadamente a imagem anômala, ou seja, a imagem gerada pelo modelo reproduz aquilo que o espaço dimensional interno, ou representação escondida, conseguiu descrever da imagem de entrada, uma vez que modelam apenas dados normais. Assim, anomalias podem ser detectadas quando há uma taxa maior no erro ao comparar as duas imagens, a imagem de entrada e a imagem gerada pelo modelo.

Isso tem mostrado alcançar resultados significativos em vários domínios. Modelos mais comuns usados são aqueles que usam um Auto-Codificador (Auto Encoder) (BERGMANN et al., 2018), (BAUR et al., 2018), onde o *encoder* realiza o mapeamento da imagem original para um espaço de características de baixa dimensão, enquanto o *decoder* tenta recuperar a informação, compressada nesse espaço, da imagem. Os parâmetros das duas redes são

aprendidos com uma função de perda baseada na reconstrução da imagem, geralmente é usada uma função $l2$ para essa tarefa.

Tal arquitetura de rede neural onde contém um gargalo, como também é conhecido o espaço de características de baixa dimensão, que é frequentemente usado para a obtenção de uma representação de condensada da imagem original, podendo ser usada também para redução de dimensionalidade. Tal gargalo força o modelo a reter as informações mais importantes para a reconstrução da imagem. Tais informações retidas devem ser tão relevantes quanto possível para as instâncias dominantes, que no caso são as instâncias normais. Como resultado, as instâncias de dados tidas como anomalias, ou seja, as que se desviam da maioria dos dados, são aquelas mal reconstruídas. O erro de reconstrução de dados pode, portanto, ser usado diretamente como pontuação de anomalia ou *anomaly score*.

Figura 18 – Demonstra estrutura de um Auto-Codificador



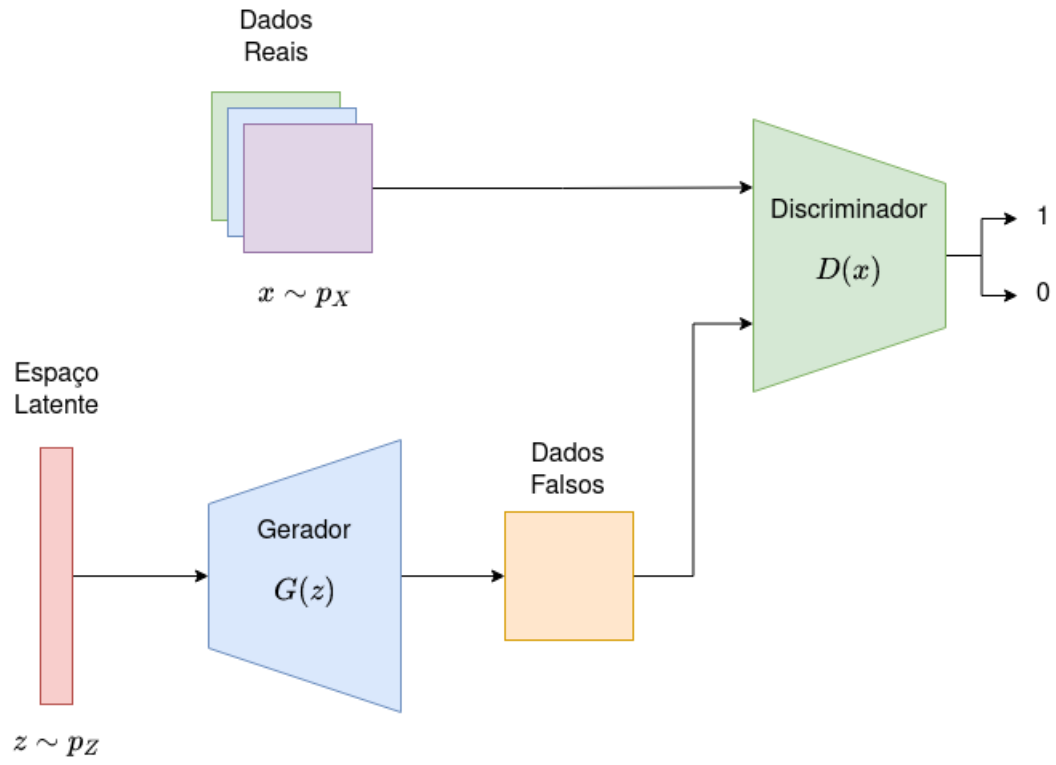
Fonte: O autor (2022)

Na Figura 18, é demonstrado como é a estrutura de um Auto Encoder, nota-se que o gargalo tem uma dimensão muito menor que a dimensão de entrada, forçando assim a rede a comprimir as informações mais relevantes para a reconstrução do dado de entrada. Mais recentemente (KWON et al., 2020) usa um Auto Codificador juntamente com a informação do gradiente em relação a algumas camadas da rede como *anomaly score*, demonstrando resultados competitivos para base de dados simples.

Entretanto, os Auto-Encoders e sua função objetivo comumente projetada para redução de dimensionalidade ou compactação de dados, em vez de detecção de anomalia. Como resultado, as representações resultantes são um resumo genérico de suas regularidades subjacentes, que não são otimizadas para detectar irregularidades. De forma geral, a imagem produzida por esse modelo tende a ser borrada com poucos detalhes quando a mesma se trata de algum objeto com muitos detalhes.

De forma similar ao Auto-Encoders, outros modelos também conseguem capturar a distribuição dos dados, como aqueles que usam redes Generative Adversarial Network (GAN). Um dos principais trabalhos onde tal rede neural foi usada foi no modelo nomeado AnoGAN (SCHLEGL et al., 2017). Em resumo AnoGAN busca uma instância z no espaço de características latente aprendido da rede generativa G, dado uma instância x, tal que a instância gerada correspondente $G(z)$ e x são tão semelhantes quanto possível. Como o espaço latente é forçado a capturar a distribuição dos dados de treinamento, espera-se que as anomalias tenham menos probabilidade de ter imagens geradas semelhantes do que as instâncias normais.

Figura 19 – Demonstra estrutura de uma rede GAN



Fonte: O autor (2022)

A rede GAN, como demonstrada na Figura 19, convencionalmente é treinada com a função objetivo:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x \sim p_X}[logD(x)] + \mathbb{E}_{z \sim p_Z}[log(1 - D(G(z)))] \tag{3.1}$$

onde G e D são respectivamente as redes geradora e o discriminadora e V é uma função valor do chamado *minmax game* (GOODFELLOW et al., 2014). Apos isso, para cada x é buscado o melhor z, através de uma busca iterativa. A busca começa com a geração aleatória de z, ao longo desse processo de busca, os parâmetros da GAN treinada são fixos, a função de perda é somente usada para atualizar os coeficientes de z a cada interação. O índice de anomalia é então definido pela similaridade de x e $G(z)$ ao término das iterações.

Apesar da abordagem AnoGAN oferecer resultados promissores e demonstrar capturar bem a distribuição dos dados de treinamento, tal abordagem é computacionalmente ineficiente, uma vez que para cada imagem há um processo de busca pelo z ótimo para a geração do índice de anomalia.

Assim como (TANG et al., 2019), ao capturar a distribuição de imagens normais, estes modelos conseguem reconstruir a imagem anômala porém sob a distribuição na qual foi treinada, ou seja, amostras normais, assim estruturas globais e locais são preservadas enquanto o modelo gera uma imagem a partir de uma outra imagem anômala que se assemelha a uma imagem normal.

Porém tais abordagens além de demandar um conjunto significativamente grande de dados, exige um maior poder de processamento computacional, tornando por muitas vezes o treinamento lento, além de que é de conhecimento comum que as redes GANs são difíceis de treinar e podem sofrer de diversos problemas, tais como falha ao convergir e o comumente chamado de *mode collapse*, onde o gerador falha em alcançar a função objetivo e os exemplos gerados são sempre os mesmos ou muito similares (METZ et al., 2016).

A abordagem Reconstruction-by-Inpainting-based Anomaly Detection (RIAD) (ZAVRTANIK; KRISTAN; SKOČAJ, 2021), por sua vez tem como base um processo comumente denominado de *Image Inpainting* (BERTALMIO et al., 2000), no qual consiste na tarefa de reconstruir partes faltantes da imagem. RIAD, usa como rede neural uma U-Net (RONNEBERGER, 2015). Apresenta resultados promissores e de sua fácil interpretabilidade, pois produz um mapa de calor para a realização da detecção e localização da anomalia. Em seu treinamento, usa o processo de *inpainting*, no qual constrói várias máscaras, onde são retiradas partes da imagem para a realização desse processo, porém, isso pode tornar o treinamento da rede neural demorado e uso crescente de memória, proporcional a quantidade de máscaras usadas. Ainda sobre RIAD, tal técnica necessita de uma quantidade grande de dados para que alcance um bom desempenho, além disso, para incorporar algum conhecimento de uma nova classe de dados, seria necessário um número significativamente grande de dados para o re-treinamento do modelo.

3.3 ABORDAGENS POR MÉTRICA

Modelos baseados em métrica constroem o modelo de maneira tal maneira que aprendam uma distância entre os dados anômalos e os dados normais. Assim como nos modelos baseados em reconstrução, abordagens baseadas em métrica podem assumir que o conjunto de treinamento é livre de anomalias. Um dos modelos mais conhecidos é Support Vector Data Description (SVDD) (TAX; DUIN, 2004), no qual força que o vetor de características, produzido pelo modelo e que representa uma imagem normal, esteja dentro de uma hiperesfera pré-definida com um raio **r** que é minimizado para conter todas as amostras de imagens normais.

Então as imagens de teste são oferecidas ao modelo, extraindo assim um vetor de características para cada imagem e caso vetor que esteja dentro de uma raio $\mathbf{r}$, tais imagens são consideradas normais, anômalas caso contrário. Trabalhos como (YI; YOON, 2020), (ZHOU et al., 2021) (RUFF et al., 2018), usam esse mesmo princípio.

Mais recentemente Patch Distribution Modeling (PaDiM) (DEFARD et al., 2020) tem demonstrado resultados promissores, no qual usa uma CNN pré-treinada juntamente com um mecanismo no qual calcula a distância de Mahalanobis (MCLACHLAN, 1999) entre as amostras de treinamento e teste baseado em pequenas janelas extraídas de diferentes camadas de uma CNN. Entretanto, o PaDiM exige para a construção de sua métrica de distância, estimar os parâmetros Gaussianos (μ_{ij}, Σ_{ij}), porém para um melhor desempenho é exigido uma quantidade de grande de dados, além de estimar Σ^{-1} que pode ser computacionalmente inviável.

3.4 ABORDAGENS POR META APRENDIZADO

Apesar de Meta Aprendizado ser usado amplamente em Aprendizagem por Reforço (WENG, 2019) e entre outros domínios, há pouca literatura que aborda o problema de detecção de anomalias em imagens.

Alguns trabalhos similares como OOD-MAML (JEONG; KIM, 2020) que identifica exemplos fora da distribuição (OOD, do Inglês: Out of Distribution) de treinamento baseado em distorções na imagem para representação de uma instância anômala, tal modelo usa o algoritmo MAML, entretanto tem como base, classificação supervisionada de tais exemplos de treinamento com distorções. Assim como o trabalho realizado por (ZHANG et al., 2020), no qual tem como objetivo a detecção de danos em rolamento em uma base de dados real em vários tipos de defeitos causado por uso contínuo, colisão e entre outros tipos de anomalias.

Já no trabalho de Lu, a detecção anomalias é realizada em câmeras de vigilância e fornece resultados promissores com relação a sua adaptação para diferentes cenários e câmeras com diferentes ângulos (LU et al., 2020). Porém ao usar uma rede GAN como modelo base, adiciona complexidade ao treinamento.

Abordagens baseadas em MANN também tem sido objeto de pesquisa na detecção de anomalias desde aplicações em bases mais simples (GONG et al., 2019), até cenário mais complexos com também em câmeras de vigilância (PARK; NOH; HAM, 2020). Entretanto, tais abordagens são dependentes do modelo e pouco flexíveis em comparação a MAML.

4 META-RIAD

Para a construção da abordagem proposta, foi visto que a abordagem RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021) apresenta resultados promissores e fornece uma maneira de inspecionar e localizar anomalias em uma imagem. Assim como também foi visto que a técnica MAML (FINN, 2017) pode ser uma solução para o problema de adaptação de novos tipos de dados, exigindo um número pequeno de amostras, por exemplo, uma ou cinco amostras de uma nova classe de dados poderiam ser o suficiente para o modelo incorporar esse conhecimento.

Sendo assim, foi observado que a combinação entre as técnicas RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021) e MAML (FINN, 2017) tem um potencial para solucionar os problemas citados. Além do desenvolvimento de uma alternativa ao método de mascaramento no processo de *inpainting*. Sendo assim a abordagem proposta foi nomeada de META-RIAD.

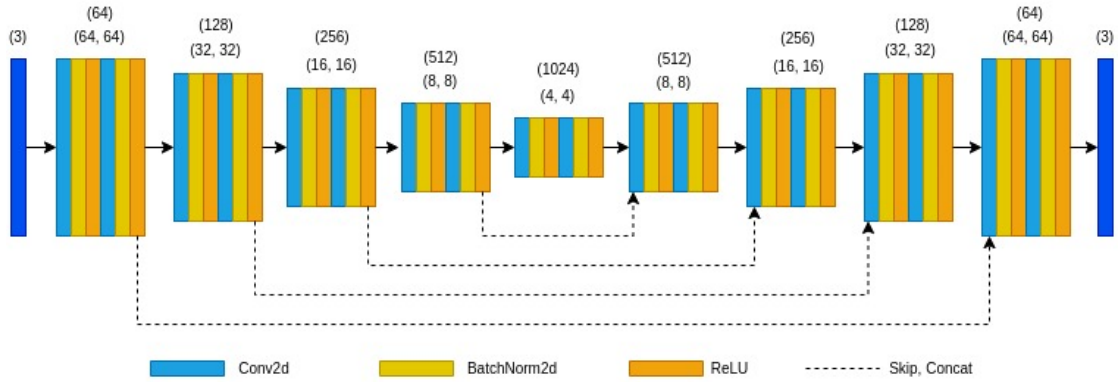
Portanto, similar ao RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021), o modelo da abordagem META-RIAD se baseia em uma arquitetura U-Net (RONNEBERGER, 2015), de nove blocos, sendo cada bloco contendo camadas convolucionais 2D com o *kernel* de tamanho 3 e padding de tamanho 1, *Batch Normalization* (IOFFE; SZEGEDY, 2015) e função de ativação Rectified Linear Unit (ReLU) (NAIR; HINTON, 2010), bem como característico da U-Net foi usado também conexões de atalho e concatenações. Todos os pesos do modelo, com exceção do bias que foi inicializado com zeros, foram inicializados com a inicialização Normal Xavier, na qual define os pesos de uma camada para valores escolhidos de uma distribuição uniforme aleatória que é limitada entre:

$$\pm\sqrt{\frac{2}{n_i + n_{i+1}}}$$

onde n_i é o número de conexões de entrada da camada, também chamado de *fan-in* e n_{i+1} é o número de conexões de saída da camada, conhecido como *fan-out*.

Como citado anteriormente, foi observado que, o processo de mascaramento da abordagem implementada por RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021), no qual gera diferentes máscaras de tamanhos distintos para cobrir diferentes partes da imagem no processo de *inpainting*, gerou uma demora significativa para o treinamento do modelo. O autor sugere que tal procedimento tem como objetivo realizar oclusões de regiões onde será feita a reconstrução da imagem, onde pode facilitar o modelo cobrir diferentes tamanhos de anomalias que o objeto possa ter. Entretanto é gerado várias máscaras disjuntas para filtrar os pixeis da imagem, sendo assim é apresentado ao modelo várias vezes a mesma imagem com diferentes máscaras a cada iteração no processo de treinamento do modelo, isso implica na construção de um grafo computacional maior, pois para cada feedforward da rede neural o algoritmo de otimização deve calcular e realizar a retropro-

Figura 20 – Diagrama do modelo U-Net usado

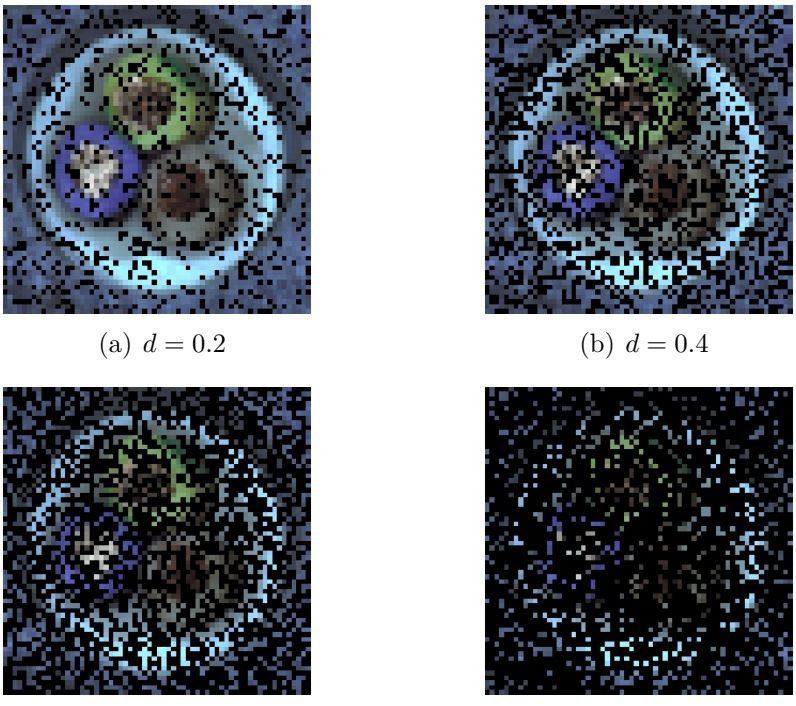


Fonte: O autor (2022)

pagação do gradiente para cada peso da rede neural, tornando assim um processamento computacionalmente custoso.

Para contornar esse problema, foi adotada uma única máscara **M** do mesmo tamanho da imagem de entrada, assim como (ZAVRTANIK; KRISTAN; SKOČAJ, 2021), a abordagem META-RIAD usa do mecanismo de *inpainting* para a remoção alguns pixeis da imagem aleatoriamente, seguindo uma determinada densidade dentro de um intervalo $[0.2, 0.8]$ como demonstrado na Figura 21, além disso a densidade de remoção também é escolhida aleatoriamente durante o treinamento.

Figura 21 – Diferentes possíveis densidades de máscaras aplicadas na imagem de entrada



(a) $d = 0.2$ (b) $d = 0.4$

(c) $d = 0.6$ (d) $d = 0.8$

Fonte: O autor (2022)

Para a realização do treinamento, foi adotado em conjunto com uma rede neural U-Net (RONNEBERGER, 2015), o algoritmo de meta-aprendizado MAML (FINN, 2017), onde se propõe a otimizar o modelo de maneira tal que com poucos exemplos de uma nova tarefa, seu desempenho seja aceitável.

Algoritmo 2: Meta Treinamento

Entrada: $p(\mathcal{T})$: distribuição sobre as tarefas
Entrada: α, β: hiper-parâmetros de taxa de aprendizado
1: aleatoriamente inicie θ
2: **enquanto** não convergir **faça**
3: Pegue K amostras do batch de tarefas $\mathcal{T}_i \sim p(\mathcal{T}) \in D_{train}$
4: **para todo** $\mathcal{T}_i = (\mathcal{D}_i^{\mathcal{S}}, \mathcal{D}_i^{\mathcal{Q}})$ **faça**
5: Avalie $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta; \mathcal{D}_i^{\mathcal{S}})$ em relação a K amostras
6: Calcule os parâmetros adaptados com gradiente descendente: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta; \mathcal{D}_i^{\mathcal{S}})$
7: **fim do para**
8: Atualize $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'; \mathcal{D}_i^{\mathcal{Q}}})$ para cada $\mathcal{D}_i^{\mathcal{Q}}$ e $\mathcal{L}_{\mathcal{T}_i}$
9: **fim do enquanto**

O treinamento do modelo segue os passos descritos no Algoritmo 2, com o objetivo de realizar com sucesso a reconstrução de uma imagem livre de anomalias. Onde D_{train}, no Algoritmo 2 é o conjunto de categorias livres de anomalias. Portanto, o modelo deve capturar a informação dos diferentes categorias livres de anomalias. Na segunda etapa, com o algoritmo MAML o modelo deve se adaptar com facilidade a novas categorias com o uso de poucos exemplos livres de anomalia, capturando assim a informação desta nova categoria em questão. Sendo assim, ao ser exposto à uma amostra do novo objeto onde contenha uma instância de anomalia, o modelo deve detectá-la.

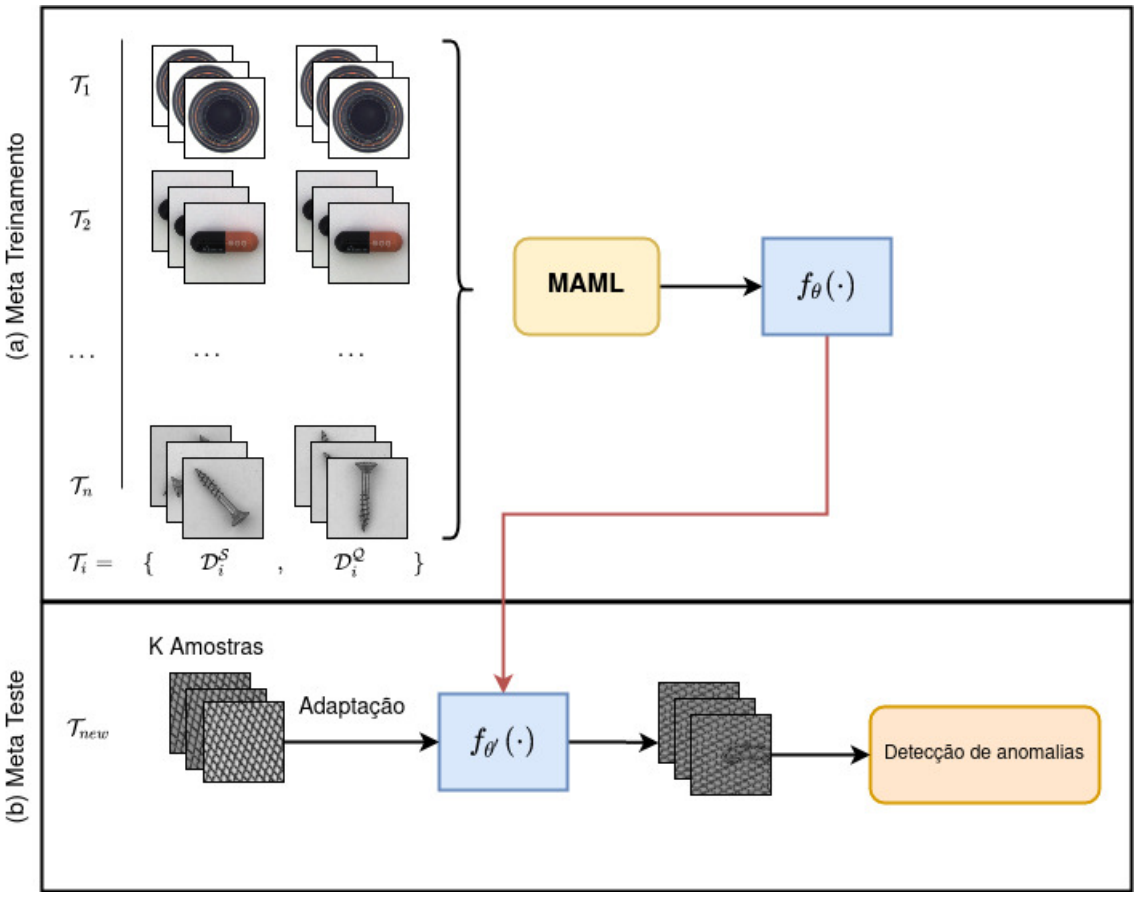
Na Figura 22 é apresentada uma visão geral na abordagem META-RIAD. No treinamento (a), são usados $M - 1$ diferentes objetos livres de anomalias. Então é usado o algoritmo de meta-aprendizado MAML para obter o modelo f_θ. Após o treinamento (b), o modelo é adaptado com K amostras do objeto que foi retirado do treinamento, também livre de anomalia, para obter um novo modelo $f_{\theta'}$. Então com o modelo adaptado $f_{\theta'}$ é realizada a detecção de anomalia no conjunto D_{test} onde contém amostras do objeto com e sem anomalias.

Similar a (LU et al., 2020), o treinamento é realizado com o conjunto D_{train}, onde $\mathcal{D}_{train} = \{\mathcal{T}_1, ..., \mathcal{T}_n\}$ e $\mathcal{T}_i = (\mathcal{D}_i^{\mathcal{S}}, \mathcal{D}_i^{\mathcal{Q}})$.

Sendo assim, o modelo é treinado para a tarefa:

$$\theta_i' = \theta - \alpha \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_\theta; \mathcal{D}_i^{\mathcal{S}}) \tag{4.1}$$

Figura 22 – Um resumo da abordagem META-RIAD



Fonte: O autor (2022)

onde

$$\mathcal{L}_{\mathcal{T}_i}(f_\theta; \mathcal{D}_i^{\mathcal{S}}) = \sum_{(x_j) \in \mathcal{D}_i^{\mathcal{S}}} L(f_\theta(x_j), x_j) \tag{4.2}$$

A loss $L(f_\theta(x_j), x_j)$ aqui mede a diferença entre a imagem de entrada e a imagem gerada pelo modelo f_θ, assim como RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021) o loss se define na combinação das funções de custo: $L2$ (L_2), $SSIM$ (L_{ssim}) (WANG et al., 2004) e uma variação em múltiplas escalas de Gradient Magnitude Similarity (GMS) (XUE et al., 2013), ou seja Multi Scale Gradient Magnitude Similarity (MSGMS) (L_{msgms}):

$$L(f_\theta(x_j), x_j) = \lambda_1 L_2 + \lambda_2 L_{ssim} + \lambda_3 L_{msgms} \tag{4.3}$$

Nos experimentos realizados os valores de λ_1, λ_2 e λ_3, assim como em RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021), foram todos iguais a 1.

A função de perda L_{msgms} é realizada primeiro calculando os mapas de magnitude de gradiente da imagem de entrada e da imagem de reconstrução, ou seja, da saída da U-Net:

$$g(\mathbf{I}) = \sqrt{(\mathbf{I} * \mathbf{h}_x)^2 + (\mathbf{I} * \mathbf{h}_y)^2} \tag{4.4}$$

onde $g(\mathbf{I})$ é o mapa de magnitude do gradiente para a imagem $\mathbf{I}$, $\mathbf{h}_x$ e $\mathbf{h}_y$ são filtros Prewitt sobre as dimensões x e y, $*$ é a operação de convolução.

O mapa GMS entre a imagem de entrada e a imagem de reconstrução então é definida como:

$$GMS(\mathbf{I}, \mathbf{I}_r) = \frac{2g(\mathbf{I})g(\mathbf{I}_r) + c}{g(\mathbf{I})^2 + g(\mathbf{I}_r)^2 + c} \tag{4.5}$$

onde a constante c assegura estabilidade numérica.

A variação em múltiplas camadas do GMS é calculada sobre uma pirâmide da imagem em quatro diferentes escalas, sendo função de perda definida como:

$$MSGMS(\mathbf{I}, \mathbf{I}_r) = \frac{1}{4}\sum_{l=1}^{4}\frac{1}{N_l}\sum_{i=1}^{H_l}\sum_{j=1}^{W_l} 1 - GMS(\mathbf{I}_l, \mathbf{I}_{rl})_{(i,j)} \tag{4.6}$$

onde H_l e W_l são respectivamente a altura e a largura da imagem, N_l é o número de pixeis na escala l. As imagens $\mathbf{I}_l$ e $\mathbf{I}_{rl}$ são a imagem de entrada e a imagem da reconstrução na escala l respectivamente. $GMS(\mathbf{I}_l, \mathbf{I}_{rl})_{(i,j)}$ é o valor do mapa GMS de $\mathbf{I}_l$ e $\mathbf{I}_{rl}$ no pixel (i, j).

Sendo assim o objetivo do meta-treinamento é encontrar a inicialização dos parâmetros θ do modelo tal que minimize:

$$\min_{\theta}\sum_{i=1}^{M}\mathcal{L}_{\mathcal{T}_i}(f_{\theta'}; \mathcal{D}_i^{\mathcal{Q}}) \tag{4.7}$$

Após o treinamento o modelo f_θ passa para fase de Meta-Teste, onde o mesmo será adaptado usando D_{adapt}. Sendo D_{adapt}, o conjunto de dados contendo somente k amostras da categoria livre de anomalias, onde o modelo será adaptado.

Assim, o Algoritmo 3 é executado, usando os conjuntos D_{adapt} e D_{test}. Sendo D_{test} o conjunto de exemplos da categoria adaptada contendo amostras com e sem anomalias.

Por fim é calculado $\varphi(\mathcal{D}_i; f_\theta)$ onde $D_{test} = \{\mathcal{D}_1, \ldots, \mathcal{D}_i\}$ sendo:

$$\varphi(\mathcal{D}_i; f_\theta) = \max(\mathbf{1}_{H\times W} - (MSGMS(\mathcal{D}_i, f_\theta(\mathcal{D}_i)) * \mathbf{f}_{sf\times sf})) \tag{4.8}$$

onde $\mathbf{f}_{sf\times sf}$ é um filtro Gaussiano de tamanho $(sf \times sf)$, $*$ é a operação de convolução e $\mathbf{1}_{H\times W}$ é uma matriz de tamanho $H \times W$. As definições detalhadas das configurações dos conjuntos D_{train}, D_{adapt} e D_{test} podem ser encontradas na Subseção 5.2.4.

Algoritmo 3: Meta Teste

Entrada: $p(\mathcal{T})$: distribuição sobre as tarefas
Entrada: α, β: hiper-parâmetros de taxa de aprendizado
Entrada: θ: parâmetros do modelo treinado com o Algoritmo 2

1: Pegue K amostras do batch de tarefas $\mathcal{T}_i \sim p(\mathcal{T}) \in D_{adapt}$
2: **para todo** $\mathcal{T}_i = (\mathcal{D}_i^{\mathcal{S}}, \mathcal{D}_i^{\mathcal{Q}})$ **faça**
3: Avalie $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta; \mathcal{D}_i^{\mathcal{S}})$ em relação a K amostras
4: Calcule os parâmetros adaptados com gradiente descendente: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta; \mathcal{D}_i^{\mathcal{S}})$
5: **fim do para**
6: Atualize $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'; \mathcal{D}_i^{\mathcal{Q}}})$ para cada $\mathcal{D}_i^{\mathcal{Q}}$ e $\mathcal{L}_{\mathcal{T}_i}$
7: **para todo** $\mathcal{D}_i \in D_{test}$ **faça**
8: Calcule $\varphi(\mathcal{D}_i; f_\theta)$ calcular o índice de anomalia para cada $\mathcal{D}_i$
9: **fim do para**

5 EXPERIMENTOS E ANÁLISE DE RESULTADOS

5.1 BASES DE DADOS

Em trabalhos mais recentes nos últimos três anos a base de dados mais utilizada para realizar a avaliação comparativa de métodos de detecção de anomalia foi a base de dados MVTec-AD (BERGMANN et al., 2019). Com o foco em inspeção industrial a base de dados contém 5354 imagens de alta resolução dividida em quinze objetos e texturas diferentes, ou seja, 3629 imagens de treinamento e 1725 imagens para teste. Cada categoria divide-se em um conjunto de treinamento com imagens sem defeitos e um conjunto de teste composto de imagens com diversos tipos de defeitos bem como imagens livre de anomalias.

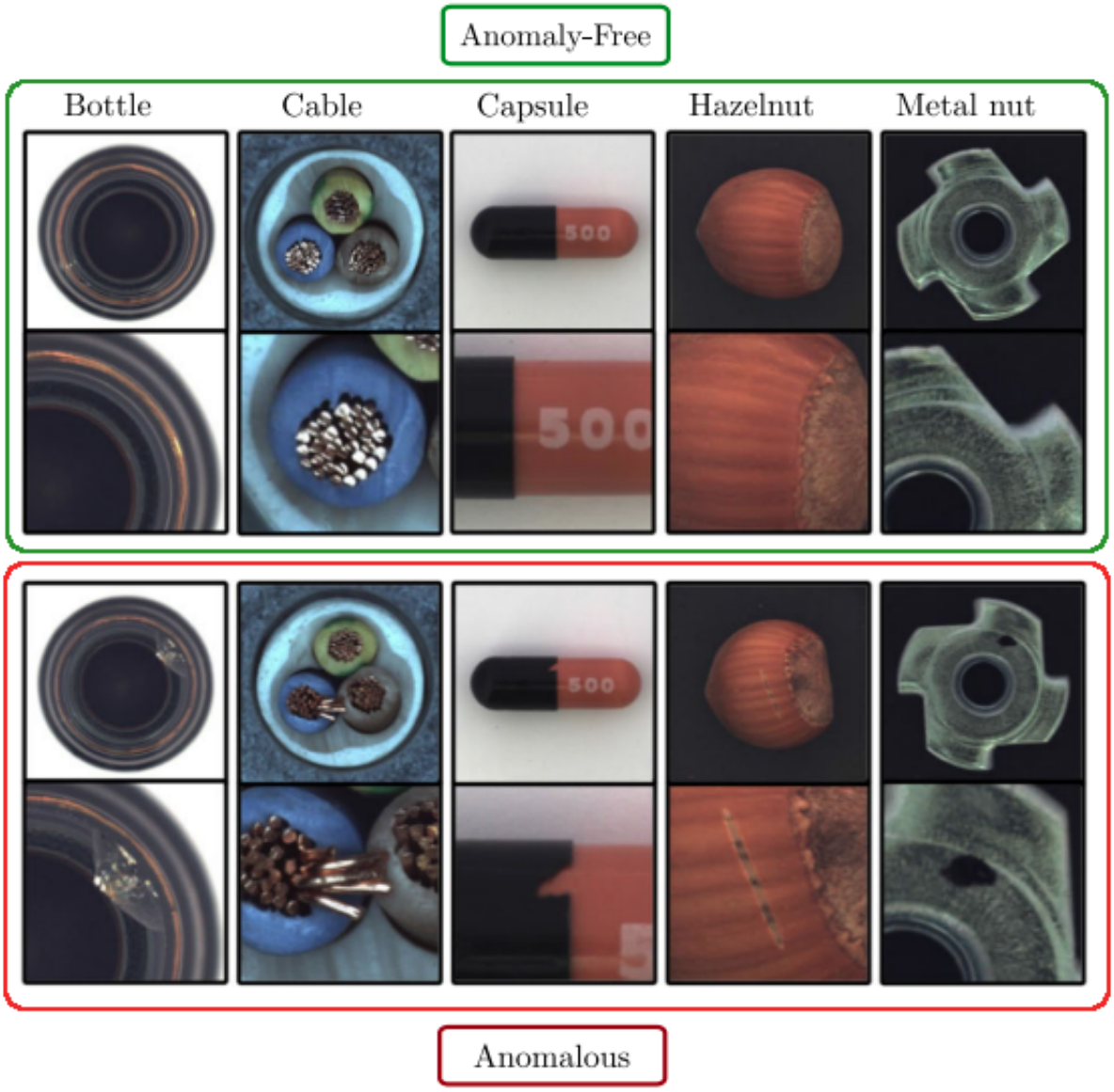
Construída manualmente, as imagens da base de dados MVTec-AD pode ser considerada aquela que se aproxima de um cenário do mundo real, onde os defeitos apresentados em cada imagem representam um defeito realmente observado no dia-a-dia. Ademais, nenhum tipo de defeito sintético foi introduzido na construção da base, ou seja, nenhum pós-processamento da imagem para a produção de exemplos com algum tipo de Data Augmentation foi realizado.

Vale ressaltar também que nenhum rótulo no conjunto de treinamento é fornecido, sendo uma boa opção para avaliar e treinar algoritmos de detecção de anomalia de maneira não supervisionada, ou seja, onde não há rótulo associado no momento do treinamento, entretanto, ao assumir a priori que o conjunto de treinamento é livre de anomalias, tal problema também pode ser considerado como semi-supervisionado.

Nas Figuras 23, 24 e 25, nota-se que cada exemplo tem uma característica diferente, seja pano de fundo, cor ou detalhes estruturais menores. Cobrindo cinco categorias de diferentes tipos de textura regular como: *Carpet* e *Grid* e textura irregular aleatória como: *Leather*, *Tile* e *Wood*. Além disso, alguns dos objetos são rígidos com uma aparência fixa como: *Bottle* e *Metal nur*, outros com estruturas deformáveis, *Calbe* ou incluindo variações naturais, *Hazelnut*. Um subconjunto de objetos estão em uma pose mais ou menos alinhada como: *Toothbrush*, *Capsule* e *Pill*, enquanto outros estão colocados de maneira frontal com uma rotação aleatória como: *Metal nut*, *Screw* e *Hazelnut*.

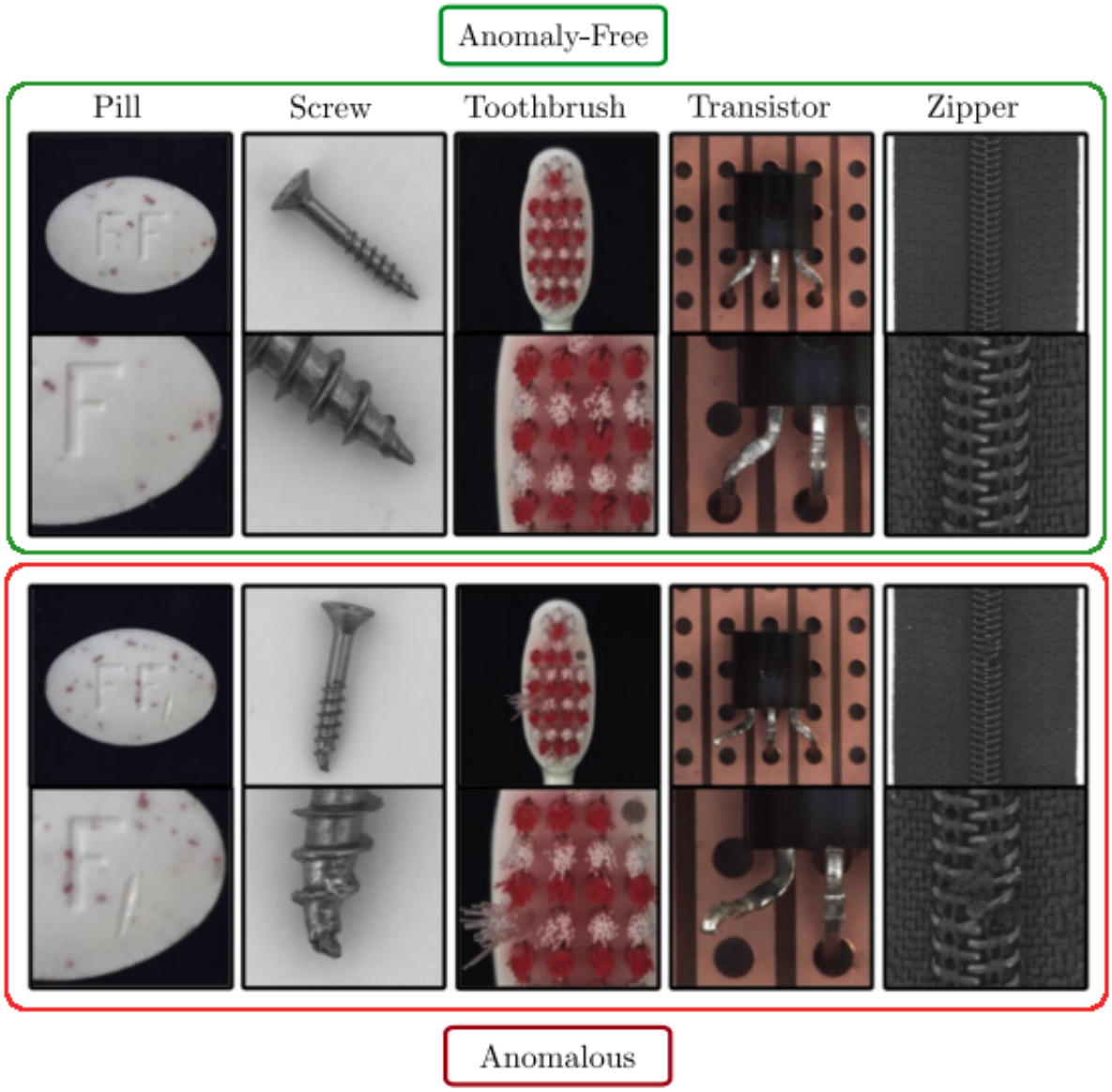
É possível também observar também na Tabela 1 uma visão global da quantidade de amostras de treinamento e teste para cada categoria, mostrando que a distribuição de amostras é diferente para cada categoria.

Figura 23 – Imagens de exemplo dos objetos Bottle, Cable, Capsule, Hazelnut e Metal nut da base de dados MVTec-AD. Para cada objeto, é mostrado um exemplo livre de anomalias e um exemplo anômalo. A linha superior mostra toda a imagem. A linha inferior oferece uma visão de perto



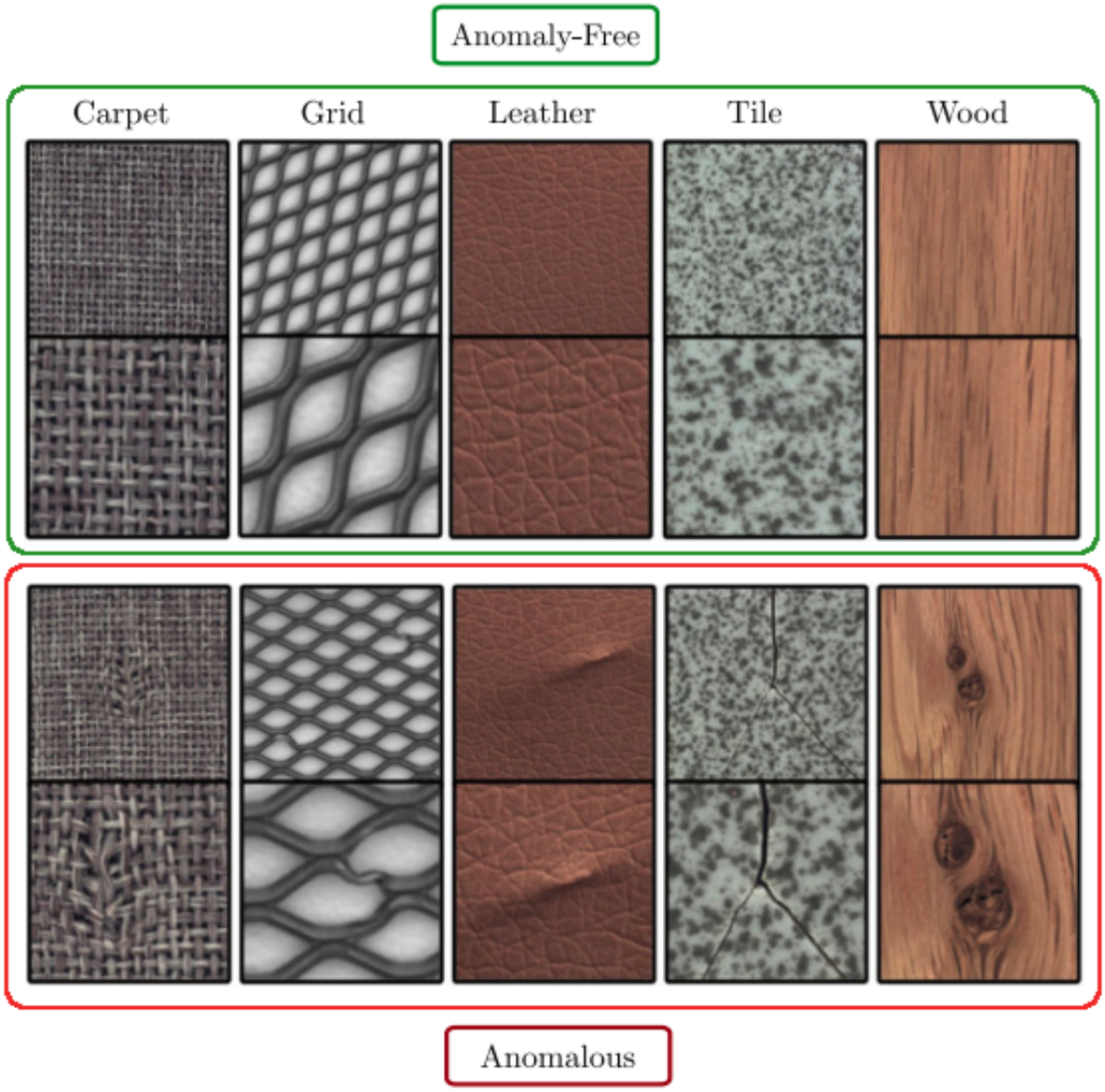
Fonte: Recorte de (BERGMANN et al., 2019)

Figura 24 – Imagens de exemplo dos objetos Pill, Screw, Toothbrush, Transistor e Zipper da base de dados MVTec-AD. Para cada objeto, é mostrado um exemplo livre de anomalias e um exemplo anômalo. A linha superior mostra toda a imagem. A linha inferior oferece uma visão de perto



Fonte: Recorte de (BERGMANN et al., 2019)

Figura 25 – Imagens de exemplo das texturas Carpet, Grid, Leather, Tile e Wood da base de dados MVTec-AD. Para cada textura, é mostrado um exemplo livre de anomalias e um exemplo anômalo. A linha superior mostra toda a imagem. A linha inferior oferece uma visão de perto



Fonte: Recorte de (BERGMANN et al., 2019)

Tabela 1 – Quantidade de imagens de treinamento e teste por categoria

	Categoria	#Treinamento	#Teste (sem anomalias)	#Teste (com anomalias)	#Total
Texturas	Carpet	280	28	89	397
	Grid	264	21	57	342
	Leather	245	32	92	369
	Tile	230	33	84	347
	Wood	247	19	60	326
Objetos	Bottle	209	20	63	292
	Cable	224	58	92	374
	Capsule	219	23	109	351
	Hazelnut	391	40	70	501
	Metal nut	220	22	93	335
	Pill	267	26	141	434
	Screw	320	41	119	480
	Toothbrush	60	12	30	102
	Transistor	213	60	40	313
	Zipper	240	32	119	391
	Total	3629	467	1258	5354

Fonte: (BERGMANN et al., 2019)

5.2 DEFINIÇÃO DOS EXPERIMENTOS

Para avaliar a capacidade dos modelos em relação ao aprendizado rápido e à adaptação a novos tipos de exemplos, similar a (LU et al., 2020), foi adaptado o protocolo sugerido por (VINYALS et al., 2016), no qual envolve fornecer ao modelo K exemplos que não foram treinados anteriormente e avaliar a capacidade do modelo de classificar novas instâncias desses exemplos fornecidos.

Para o presente trabalho, os modelos foram pré-treinados com $M - 1$ categorias, por vez, onde tais categorias neste momento são livres de anomalias, ou seja, os modelos não tem acesso ao que foi retirado no treinamento, bem como a qualquer amostra dos itens contidas no conjunto de treinamento que contenha alguma anomalia. No momento da avaliação são fornecidos ao modelo K exemplos de um novo tipo de categoria, livre de anomalias e posteriormente é apresentado aos modelos o conjunto de teste da categoria apresentada no início da avaliação, ou seja, neste conjunto de teste existem exemplos da categoria com e sem anomalias. Podemos denominar nesse tipo de protocolo como *K-shots Anomaly Detection*, pois é realizado a detecção de anomalia em imagens de uma nova categoria com somente K amostras de treinamento.

Como não houve até o momento uma abordagem que tenha uma configuração experimental prévia à desenvolvida nesta pesquisa, para validar o desempenho do modelo proposto de maneira consistente, foram considerados os seguintes algoritmos como baselines para comparação: AnoGrad (KWON et al., 2020), RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021) e PaDiM (DEFARD et al., 2020). Cada modelo foi pré-treinado conforme a seção a seguir, de maneira que todos os modelos tenham sido expostos as condições similares no que no que se refere a conjunto de dados.

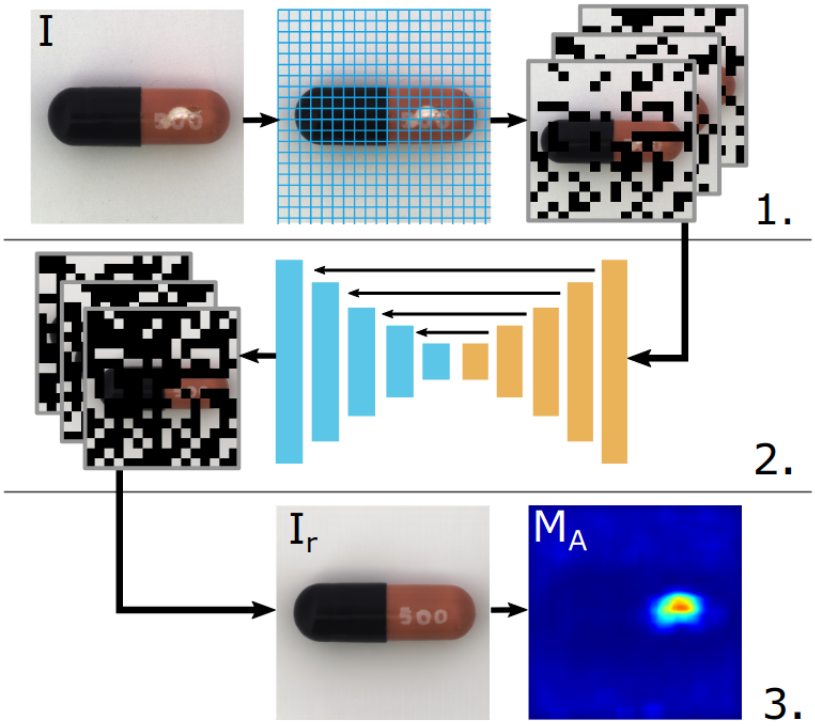
Nas seções a seguir são apresentados de maneira geral as baselines adotadas, onde no presente momento da execução dos experimentos deste trabalho, são os modelos considerados estado da arte, onde PaDiM e RIAD na base de dados MVTec-AD (BERGMANN et al., 2019) e AnoGrad nas bases de dados CIFAR-10 (KRIZHEVSKY; NAIR; HINTON, 2014) e MNIST (DENG, 2012).

5.2.1 RIAD: Reconstruction by inpainting for visual anomaly detection

Como mostra a Figura 26, a abordagem RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021), consiste em repartir uma imagem $\mathbf{I}$ em um grid de regiões retangulares de tamanho $k \times k$ pixeis. Tais conjuntos de regiões são aleatoriamente separadas em n subconjuntos disjuntos, assim para cada subconjunto, as regiões pertencentes a esse subconjunto são removidas da imagem original, resultando em n imagens de entrada.

As imagens de entrada são reconstruídas usando uma rede U-Net (RONNEBERGER, 2015) realizando o processo de *inpainting*, gerando assim n imagens de saída, cada uma

Figura 26 – Diagrama ilustrando a técnica RIAD



Fonte: (ZAVRTANIK; KRISTAN; SKOČAJ, 2021)

reconstruindo as regiões removidas em sua imagem de entrada correspondente. As regiões reconstruídas individuais das n reconstruções parciais são reagrupadas em uma única imagem reconstruída $\mathbf{I}_r$, então a reconstrução da imagem é avaliada e produzido o índice de anomalia baseado em uma variação em múltiplas escalas de GMS (XUE et al., 2013), onde também é produzido um mapa de calor M_A com a estimativa da localização da anomalia na imagem.

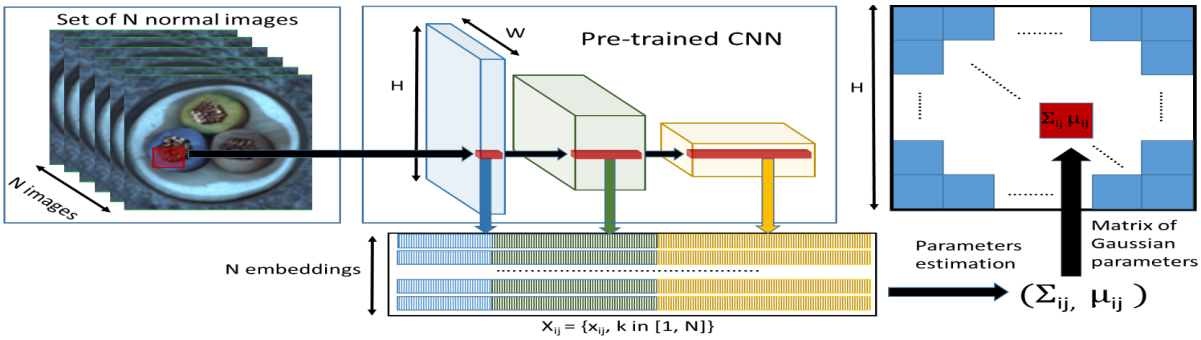
5.2.2 PaDiM: Patch Distribution Modeling Framework for Anomaly Detection and Localization

Mostrada na Figura 27, a abordagem PaDiM (DEFARD et al., 2020), faz uso de uma CNN pré-treinada para a extração de pedaços da imagem em forma de *patch*, onde para cada um é gerado um vetor de *embedding*, ou seja, para cada posição do *patch* na imagem é produzido um vetor de *embedding* x_{ij}.

Para extrair o conhecimento das imagens ditas como normais na posição (i, j), primeiro é calculado o conjunto de vetores de *embedding* dos *patches* na posição (i, j) de N imagens de treinamento, $X_{ij} = \{x_{ij}^k, k \in [1, N]\}$.

Assim, o modelo assume que X_{ij} é gerado por uma distribuição gaussiana multivariada

Figura 27 – Diagrama demonstrando a técnica PaDiM



Fonte: (DEFARD et al., 2020)

$\mathcal{N}(\mu_{ij}, \Sigma_{ij})$ onde μ_{ij} é a média amostral de X_{ij} e Σ_{ij} a covariância amostral estimada por:

$$\Sigma_{ij} = \frac{1}{N-1}\sum_{k=1}^{N}(x_{ij}^k - \mu_{ij})(x_{ij}^k - \mu_{ij})^T + \epsilon I \tag{5.1}$$

onde o terno de regularização ϵI faz a matriz amostral de covariância invertível.

Por fim, para a produção o modelo usa a distância de Mahalanobis, $M(x_{ij}))$, para calcular índice de anomalia na posição do *patch* (i, j) de uma imagem de teste. Assim $M(x_{ij}))$ é interpretado como uma distância entre o vetor de *embedding* do *patch* x_{ij} e a distribuição aprendida $\mathcal{N}(\mu_{ij}, \Sigma_{ij})$, onde $M(x_{ij})$ é calculado pela Equação 5.2:

$$M(x_{ij}) = \sqrt{(x_{ij} - \mu_{ij})^T \Sigma_{ij}^{-1}(x_{ij} - \mu_{ij})} \tag{5.2}$$

Assim o índice final de anomalia é o valor máximo entre todos os valores de $M(x_{ij})$.

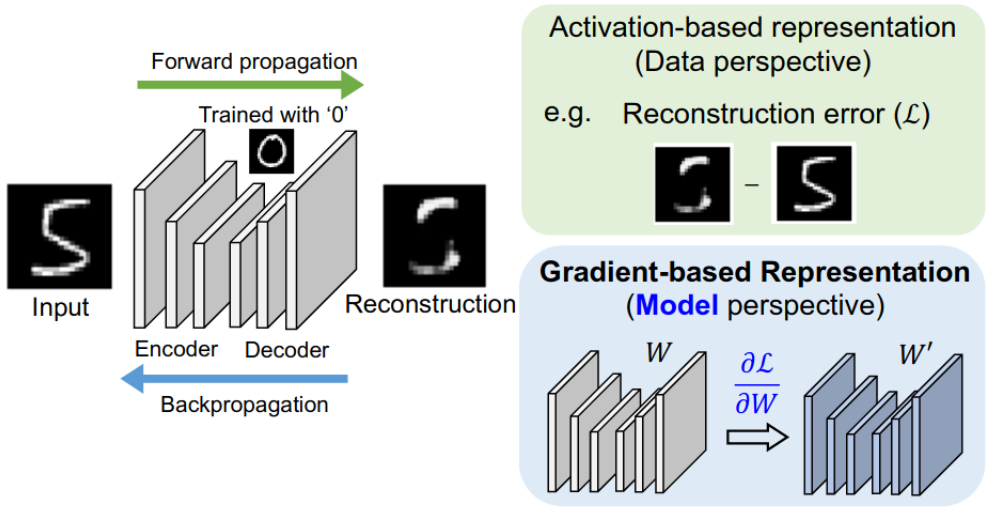
5.2.3 AnoGrad: Backpropagated Gradient Representations for Anomaly Detection

Como mostra na Figura 28, de modo geral a abordagem AnoGrad (KWON et al., 2020), usa representações baseadas no gradiente para a detecção de anomalias ao descrever as atualizações do modelo causadas por dados. Usando uma rede neural Auto Encoder, o modelo adiciona à função de perda $\mathcal{J}$ uma outra função de perda denominada $\mathcal{L}_{grad}$ como termo de regularização.

Tal função é calculada com a similaridade do coseno entre os gradientes de uma determinada camada i do componente da rede o *Decoder* na k ésima interação do treinamento, $\frac{\partial \mathcal{L}}{\partial \phi_i}^k$ e a média dos gradientes do treinamento da mesma camada i obtida até a $k-1$ ésima interação, $\frac{\partial \mathcal{J}}{\partial \phi_i}_{avg}^{k-1}$. A função de perda $\mathcal{L}_{grad}$ na k ésima interação do treinamento é obtida realizando a média da similaridade do cosseno sobre todas as camadas do *Decoder*:

$$\mathcal{L}_{grad} = -\mathbb{E}_i[cosSIM(\frac{\partial \mathcal{J}}{\partial \phi_i}_{avg}^{k-1}, \frac{\partial \mathcal{L}}{\partial \phi_i}^k)], \qquad \frac{\partial \mathcal{J}}{\partial \phi_i}_{avg}^{k-1} = \frac{1}{(k-1)}\sum_{t=1}^{k-1}\frac{\partial \mathcal{J}^t}{\partial \phi_i} \tag{5.3}$$

Figura 28 – Diagrama demonstrando a técnica AnoGrad



Fonte: (KWON et al., 2020)

onde J é definida como $J = \mathcal{L} + \Omega + \alpha\mathcal{L}_{grad}$. O primeiro e o segundo termos são o erro de reconstrução e a perda latente, respectivamente, e são definidos por diferentes tipos de autoencoders. α é um peso para a função de perda $\mathcal{L}_{grad}$.

Durante o treinamento, $\mathcal{L}$ é primeiro calculado da propagação *forward*. Atravéz da retro-propagação, $\frac{\partial\mathcal{L}}{\partial\phi_i}^k$ é obtida sem a atualização dos pesos da rede. Então, baseado no gradiente obtido, toda a função de perda J é calculada e os pesos são atualizados usando a retro-propagação dos gradientes. O índice de anomalia é definido pela combinação do erro da reconstrução e o valor da função de perda $\mathcal{L}_{grad}$: $\mathcal{L} + \beta\mathcal{L}_{grad}$, onde β o autor define como $\beta = 4\alpha$.

5.2.4 Protocolo de Avaliação

Para que seja possível avaliar as técnicas supracitadas e a abordagem META-RIAD com relação a sua capacidade de adaptação a um tipo um novo de exemplo nunca visto, bem como a sua capacidade de incorporar conhecimento com um número de amostras restrito, se faz necessário que haja subconjuntos de dados onde: o modelo adquira conhecimento anterior, adicione o conhecimento novo com um numero limitado de amostras e seja avaliado com relação ao conhecimento recentemente incorporado de acordo com a métrica definida.

Portanto, similar a (LU et al., 2020), foram adotados três conjuntos da base MVTec-AD nomeados como D_{train}, D_{adapt} e D_{test}, sendo D_{train} o conjunto de categorias livre de anomalias, porém é retirado uma categoria, ficando assim $M - 1$ categorias, o conjunto D_{adapt} que é a categoria retirada, esta também é livre de anomalias, onde o modelo vai

realizar a adaptação com K amostras, por fim D_{test} é o conjunto de amostras da categoria retirada porém contem instâncias sem e com anomalia com o propósito de realizar a avaliação da técnica em questão.

De maneira geral, como proposto por (VINYALS et al., 2016), o problema de classificação conhecido como *N-Way K-Shot* se refere a construção de tarefas com um número de amostras pequeno. Ao construir cada tarefa, N classes são extraídas do conjunto de dados e cada classe é composta de K amostras cada. Assim, cada tarefa imita o cenário conhecido como *few shot*, sendo divido no conjunto de suporte (*support set*) que é usado para o modelo aprender como resolver tal tarefa. Ademais, outros exemplos das mesmas classes são obtidos para a avaliação da performance nesta tarefa em particular, esse conjunto também é conhecido como conjunto de consulta ou *query set*.

Cada tarefa pode ser completamente não sobreposta, ou seja, o modelo pode nunca ver as classes de uma tarefa em nenhuma das outras. A ideia é que o modelo repetidamente veja instancias (tarefas) durante o treinamento que se associem a estrutura da tarefa final de *few shot*, porém contendo classes diferentes.

Para o presente trabalho, foi adotada uma estrutura similar de treinamento, entretanto para a abordagem META-RIAD foi usada somente uma categoria por tarefa, tornando assim a construção das mesmas em *1-way k-shot*, onde no presente trabalho é referido somente como *k-shot*. Vale ressaltar que no decorrer do treinamento $M-1$ categorias são obtidas no treinamento e a categoria que foi retirada do treinamento é exposta na fase de Meta-Teste, como pode-se ver na Figura 22.

Como exemplo, ao avaliar a capacidade dos modelos na incorporação da classe *Cable* na tarefa de detecção de anomalias com um número de amostras limitada. É separado o conjunto contendo todas as classes da base de dados MVTec-AD menos o objeto *Cable*, tal conjunto contem somente amostras ditas como normais, tal conjunto aqui é denominado como D_{train}. Então, outro conjunto contendo amostras livre de anomalias, agora do objeto *Cable* é selecionado, esse conjunto é denominado D_{adapt}, onde cada modelo irá tentar incorporar o conhecimento desse particular objeto com somente K amostras. Por fim, para avaliar se o modelo, agora adaptado com o conhecimento do objeto livre de anomalias *Cable*, é selecionado um terceiro conjunto contendo amostras com e sem anomalias do objeto *Cable*, nomeado de D_{test}, neste momento é realizada a detecção de anomalias neste objeto em questão e avaliado conforme a métrica definida.

Sendo assim para analisar a capacidade de adaptação das abordagens, em diferentes quantidades de amostras fornecidas as abordagens, cada uma foi aplicada nas configurações de *1-shot*, *5-shots* e *10-shots* na detecção de anomalias.

Na tarefa de detecção de anomalias a métrica mais comum usada é a Area Under the Receiver Operating Characteristic Curve (AUC-ROC), ela mede a qualidade das previsões do modelo, independentemente do limiar de classificação escolhido, a AUC-ROC varia em valor de 0 a 1. Um modelo cujas previsões estão 100% erradas tem uma AUC-ROC de

0.0, aquele cujas previsões estão 100% corretas tem uma AUC-ROC de 1.0, sendo assim foi adotada neste trabalho para reportar os resultados.

5.2.5 Pré-Treinamento

Similar a (LU et al., 2020), para a comparação dos resultados com outros modelos, os mesmos foram pré-treinados na base MVTec-AD. De maneira geral o pré-treinamento dos modelos foi realizado com todos os objetos com exceção de um objeto em particular, então após o pré-treinamento foi utilizado o modelo pré-treinado, em novo conjunto contendo o objeto que foi retirado no pre-treinamento que agora está presente, ou seja, ele é pré-treinado no conjunto onde existem $M - 1$ categorias e o treinamento final somente com a categoria que foi retirada. Sendo assim no treinamento final o modelo, nunca foi apresentado o objeto no qual houve um pré-treinamento anterior. Por fim é avaliado no conjunto da categoria contendo imagens com e sem anomalias.

5.2.6 Ambiente

Para a realização dos experimentos, foi utilizado uma infraestrutura adequada listada na Tabela 2, contando com uma *gpu* que tornou possível a execução, mesmo que demorada, de todos os experimentos de todos os modelos e cenários planejados na base de dados escolhida.

Tabela 2 – Infra-estrutura utilizada para realização dos experimentos

Framework	Pytorch 1.8.2
CUDA	11.4
Linguagem de Programação	Python 3.8.10
GPU	NVIDIA GeForce RTX 2060
Memória GPU	6 GB
Armazenamento	512 GB
Processador	Intel® Core™ i7-10750H CPU @ 2.60GHz × 12
Memória RAM	16 GB

Fonte: O autor (2022)

5.3 RESULTADOS

Para uma melhor análise e interpretação dos resultados, os mesmos foram subdivididos em texturas e objetos, bem como para cada *K-shot* observado. Cada modelo foi treinado e avaliado com cinco *seeds* aleatórias diferentes, sendo assim os resultados reportados são a média entre as execuções, bem como são reportados os desvios padrões de cada.

5.3.1 Análise Quantitativa

Tabela 3 – Resultados dos modelos para *1-shot* na métrica AUC-ROC (objetos)

Objeto	RIAD	PADIM	AnoGrad	META-RIAD
bottle	0.933 ± 0.014	0.882 ± 0.008	0.285 ± 0.067	**0.950 ± 0.006**
cable	0.635 ± 0.010	0.627 ± 0.005	0.453 ± 0.082	**0.642 ± 0.015**
capsule	0.489 ± 0.046	**0.577 ± 0.008**	0.517 ± 0.073	0.500 ± 0.010
hazelnut	0.663 ± 0.019	0.423 ± 0.031	0.084 ± 0.031	**0.685 ± 0.003**
metal_nut	0.493 ± 0.098	0.399 ± 0.025	**0.725 ± 0.077**	0.600 ± 0.002
pill	0.575 ± 0.018	0.494 ± 0.004	0.349 ± 0.084	**0.618 ± 0.008**
screw	0.481 ± 0.164	**0.967 ± 0.010**	0.030 ± 0.139	0.501 ± 0.009
toothbrush	0.661 ± 0.051	0.756 ± 0.009	0.525 ± 0.054	**0.780 ± 0.005**
transistor	0.662 ± 0.082	0.679 ± 0.007	0.484 ± 0.067	**0.770 ± 0.011**
zipper	0.588 ± 0.103	0.652 ± 0.004	**0.681 ± 0.062**	0.671 ± 0.009

Fonte: O autor (2022)

Como podemos ver na Tabela 3, na tarefa de detecção de anomalias em imagens de objetos com a configuração de *1-shot*, a abordagem META-RIAD teve um desempenho superior nos 7 dos 10 casos avaliados, seguido do modelo PaDiM e AnoGrad que ambos obtiveram melhor desempenho em 2 dos 10 casos avaliados.

Tabela 4 – Resultados dos modelos para *1-shot* na métrica AUC-ROC (texturas)

Textura	RIAD	PaDiM	AnoGrad	META-RIAD
carpet	0.678 ± 0.039	0.324 ± 0.021	0.089 ± 0.175	**0.690 ± 0.011**
grid	0.480 ± 0.189	0.150 ± 0.017	0.441 ± 0.161	**0.510 ± 0.006**
leather	**0.666 ± 0.021**	0.396 ± 0.012	0.334 ± 0.092	0.650 ± 0.015
tile	0.811 ± 0.003	0.742 ± 0.005	0.429 ± 0.139	**0.831 ± 0.004**
wood	0.844 ± 0.010	0.733 ± 0.021	0.252 ± 0.062	**0.887 ± 0.027**

Fonte: O autor (2022)

A Tabela 4 apresenta o desempenho dos modelos na detecção de anomalias em texturas, a abordagem META-RIAD também apresentou resultados superiores em 4 dos 5 casos avaliados na configuração de *1-shot*. É importante notar aqui, que desta vez o modelo RIAD obteve melhor desempenho em relação aos modelos PaDiM e AnoGrad, sendo melhor em 1 dos 5 casos avaliados.

De maneira geral, como evidenciado na Tabela 5 para a tarefa de detecção de anomalia em imagens de objetos e texturas, na configuração de *1-shot*, a abordagem META-RIAD conseguiu obter o melhor resultado em 10 dos 15 casos avaliados em relação aos demais modelos.

A Tabela 6 apresenta os resultados dos modelos na detecção de anomalias em imagens de objetos com a configuração de *5-shot*, aqui a abordagem META-RIAD se destaca em 4

Tabela 5 – Resultados dos modelos para *1-shot* na métrica AUC-ROC (objetos e texturas)

Item	RIAD	PaDiM	AnoGrad	META-RIAD
bottle	0.933 ± 0.014	0.882 ± 0.008	0.285 ± 0.067	**0.950 ± 0.006**
cable	0.635 ± 0.010	0.627 ± 0.005	0.453 ± 0.082	**0.642 ± 0.015**
capsule	0.489 ± 0.046	**0.577 ± 0.008**	0.517 ± 0.073	0.500 ± 0.010
carpet	0.678 ± 0.039	0.324 ± 0.021	0.089 ± 0.175	**0.690 ± 0.011**
grid	0.480 ± 0.189	0.150 ± 0.017	0.441 ± 0.161	**0.510 ± 0.006**
hazelnut	0.663 ± 0.019	0.423 ± 0.031	0.084 ± 0.031	**0.685 ± 0.003**
leather	**0.666 ± 0.021**	0.396 ± 0.012	0.334 ± 0.092	0.650 ± 0.015
metal_nut	0.493 ± 0.098	0.399 ± 0.025	**0.725 ± 0.077**	0.600 ± 0.002
pill	0.575 ± 0.018	0.494 ± 0.004	0.349 ± 0.084	**0.618 ± 0.008**
screw	0.481 ± 0.164	**0.967 ± 0.010**	0.030 ± 0.139	0.501 ± 0.009
tile	0.811 ± 0.003	0.742 ± 0.005	0.429 ± 0.139	**0.831 ± 0.004**
toothbrush	0.661 ± 0.051	0.756 ± 0.009	0.525 ± 0.054	**0.780 ± 0.005**
transistor	0.662 ± 0.082	0.679 ± 0.007	0.484 ± 0.067	**0.770 ± 0.011**
wood	0.844 ± 0.010	0.733 ± 0.021	0.252 ± 0.062	**0.887 ± 0.027**
zipper	0.588 ± 0.103	0.652 ± 0.004	**0.681 ± 0.062**	0.671 ± 0.009

Fonte: O autor (2022)

Tabela 6 – Resultados dos modelos para *5-shots* na métrica AUC-ROC (objetos)

Objeto	RIAD	PaDiM	AnoGrad	META-RIAD
bottle	**0.965 ± 0.005**	0.898 ± 0.007	0.640 ± 0.010	0.963 ± 0.018
cable	0.746 ± 0.059	0.696 ± 0.011	0.374 ± 0.016	**0.834 ± 0.005**
capsule	0.526 ± 0.015	**0.583 ± 0.005**	0.542 ± 0.019	0.539 ± 0.017
hazelnut	0.664 ± 0.010	0.352 ± 0.010	0.155 ± 0.083	**0.690 ± 0.009**
metal_nut	0.551 ± 0.031	0.439 ± 0.031	**0.820 ± 0.043**	0.678 ± 0.009
pill	0.544 ± 0.048	**0.656 ± 0.015**	0.409 ± 0.057	0.648 ± 0.018
screw	0.079 ± 0.014	**0.712 ± 0.013**	0.090 ± 0.044	0.410 ± 0.002
toothbrush	0.877 ± 0.050	**0.963 ± 0.011**	0.586 ± 0.041	0.884 ± 0.019
transistor	0.705 ± 0.031	0.703 ± 0.018	0.464 ± 0.023	**0.789 ± 0.012**
zipper	0.623 ± 0.141	0.670 ± 0.002	0.462 ± 0.044	**0.705 ± 0.007**

Fonte: O autor (2022)

dos 10 casos avaliados, assim como o modelo PaDiM que também se destacou em 4 casos de 10 avaliados.

Vale notar que, em relação configuração *1-shot* ao comparar os resultados vistos na Tabela 3, onde é apresentado o desempenho das abordagens com relação aos objetos, houve um aumento no desempenho da abordagem META-RIAD, evidenciando assim que ao aumentar a quantidade de amostras oferecidas ao modelo, há um aumento no desempenho do mesmo.

Na Tabela 7, mostra que a abordagem META-RIAD tem um desempenho superior em 5 casos dos 5 avaliados, na detecção de anomalias em texturas, sendo o melhor resultado obtido em comparação aos demais.

De maneira similar, houve um aumento no desempenho em relação a configuração

Tabela 7 – Resultados dos modelos para *5-shots* na métrica AUC-ROC (texturas)

Textura	RIAD	PaDiM	AnoGrad	META-RIAD
carpet	0.629 ± 0.017	0.468 ± 0.029	0.103 ± 0.037	**0.650 ± 0.011**
grid	0.695 ± 0.019	0.356 ± 0.012	0.430 ± 0.091	**0.710 ± 0.004**
leather	0.576 ± 0.074	0.500 ± 0.007	0.335 ± 0.103	**0.662 ± 0.007**
tile	0.798 ± 0.025	0.773 ± 0.006	0.333 ± 0.071	**0.849 ± 0.006**
wood	0.866 ± 0.010	0.780 ± 0.016	0.266 ± 0.024	**0.920 ± 0.015**

Fonte: O autor (2022)

1-shot ao comparar os resultados vistos na Tabela 4, onde apresenta os resultados com relação as texturas.

Tabela 8 – Resultados dos modelos para *5-shots* na métrica AUC-ROC (objetos e texturas)

Item	RIAD	PaDiM	AnoGrad	META-RIAD
bottle	**0.965 ± 0.005**	0.898 ± 0.007	0.640 ± 0.010	0.963 ± 0.018
cable	0.746 ± 0.059	0.696 ± 0.011	0.374 ± 0.016	**0.834 ± 0.005**
capsule	0.526 ± 0.015	**0.583 ± 0.005**	0.542 ± 0.019	0.539 ± 0.017
carpet	0.629 ± 0.017	0.468 ± 0.029	0.103 ± 0.037	**0.650 ± 0.011**
grid	0.695 ± 0.019	0.356 ± 0.012	0.430 ± 0.091	**0.710 ± 0.004**
hazelnut	0.664 ± 0.010	0.352 ± 0.010	0.155 ± 0.083	**0.690 ± 0.009**
leather	0.576 ± 0.074	0.500 ± 0.007	0.335 ± 0.103	**0.662 ± 0.007**
metal_nut	0.551 ± 0.031	0.439 ± 0.031	**0.820 ± 0.143**	0.678 ± 0.009
pill	0.544 ± 0.048	**0.656 ± 0.015**	0.409 ± 0.057	0.648 ± 0.018
screw	0.079 ± 0.014	**0.712 ± 0.013**	0.090 ± 0.044	0.410 ± 0.002
tile	0.798 ± 0.025	0.773 ± 0.006	0.333 ± 0.071	**0.849 ± 0.006**
toothbrush	0.877 ± 0.050	**0.963 ± 0.011**	0.586 ± 0.041	0.884 ± 0.019
transistor	0.705 ± 0.031	0.703 ± 0.018	0.464 ± 0.023	**0.789 ± 0.012**
wood	0.866 ± 0.010	0.780 ± 0.016	0.266 ± 0.024	**0.920 ± 0.015**
zipper	0.623 ± 0.141	0.670 ± 0.002	0.462 ± 0.044	**0.705 ± 0.007**

Fonte: O autor (2022)

De modo geral, como visto na Tabela 8, para a tarefa de detecção de anomalia em objetos e texturas, na configuração de *5-shots*, a abordagem META-RIAD apresentou bom desempenho, se destacando os 9 casos dos 15 avaliados. Também notou-se que, em relação a configuração *1-shot*, os modelos avaliados apresentaram aumento no desempenho, com destaque a abordagem META-RIAD, evidenciando assim que ao aumentar a quantidade de *k-shots* os modelos tem um aumento na performance observada.

Na Tabela 9 mostra os resultados para a tarefa de detecção de anomalia em imagens de objetos na configuração *10-shots*, mais uma vez a abordagem META-RIAD se destaca em 6 casos de 10 avaliados. Em relação aos resultados apresentados na Tabela 6, onde mostra os resultados obtidos na configuração *5-shots* com as imagens de objetos, o crescimento do desempenho dos modelos é mantido.

Como visto na Tabela 10, a abordagem META-RIAD apresentou resultados superiores

Tabela 9 – Resultados dos modelos para *10-shots* na métrica AUC-ROC (objetos)

Objeto	RIAD	PaDiM	AnoGrad	META-RIAD
bottle	0.971 ± 0.006	0.908 ± 0.011	0.649 ± 0.010	**0.982 ± 0.003**
cable	0.740 ± 0.011	0.698 ± 0.005	0.378 ± 0.032	**0.830 ± 0.007**
capsule	0.531 ± 0.015	**0.606 ± 0.004**	0.563 ± 0.013	0.540 ± 0.005
hazelnut	0.697 ± 0.008	0.452 ± 0.011	0.517 ± 0.014	**0.720 ± 0.003**
metal_nut	0.562 ± 0.018	0.377 ± 0.013	0.264 ± 0.058	**0.684 ± 0.023**
pill	0.549 ± 0.072	**0.704 ± 0.014**	0.487 ± 0.061	0.672 ± 0.002
screw	0.192 ± 0.011	**0.699 ± 0.025**	0.128 ± 0.036	0.575 ± 0.013
toothbrush	0.824 ± 0.024	**0.960 ± 0.003**	0.525 ± 0.031	0.908 ± 0.004
transistor	0.690 ± 0.013	0.727 ± 0.015	0.705 ± 0.056	**0.834 ± 0.004**
zipper	0.719 ± 0.010	0.729 ± 0.004	0.662 ± 0.019	**0.778 ± 0.004**

Fonte: O autor (2022)

Tabela 10 – Resultados dos modelos para *10-shots* na métrica AUC-ROC (texturas)

Textura	RIAD	PaDiM	AnoGrad	META-RIAD
carpet	**0.750 ± 0.014**	0.470 ± 0.022	0.522 ± 0.036	0.663 ± 0.014
grid	0.714 ± 0.041	0.396 ± 0.024	0.460 ± 0.106	**0.786 ± 0.013**
leather	0.665 ± 0.012	0.521 ± 0.003	0.327 ± 0.057	**0.688 ± 0.010**
tile	0.834 ± 0.016	0.775 ± 0.005	0.405 ± 0.063	**0.853 ± 0.002**
wood	0.897 ± 0.029	0.808 ± 0.005	0.373 ± 0.034	**0.941 ± 0.003**

Fonte: O autor (2022)

em 4 dos 5 casos avaliados na configuração de *10-shots* para texturas, obtendo o melhor resultado em relação aos demais. Da mesma maneira, houve um aumento no desempenho em relação aos resultados obtidos nas texturas na configuração *5-shots* como podem ser comparados na Tabela 7.

Em geral, como evidenciado na Tabela 11, onde apresenta os resultados obtidos nas texturas e objetos na configuração *10-shots*, a abordagem META-RIAD apresentou melhores resultados em 10 dos 15 casos avaliados, obtendo um melhor resultados em relação aos demais modelos. Assim como nos resultados obtidos na configuração *5-shots* na detecção de anomalias em objetos e texturas visto na Tabela 8, o aumento no desempenho médio dos modelos continua.

Foi observado também que, em alguns casos seja objeto ou textura, ao aumentar a quantidade de *shots*, não há melhora significativa nos resultados, como exemplo o modelo PaDiM que na configuração *5-shots* no objeto *Toothbrush*, não houve melhora significativa para a configuração de *10-shots*, sendo os resultados saíram de 0.963 ± 0.011 para 0.960 ± 0.003.

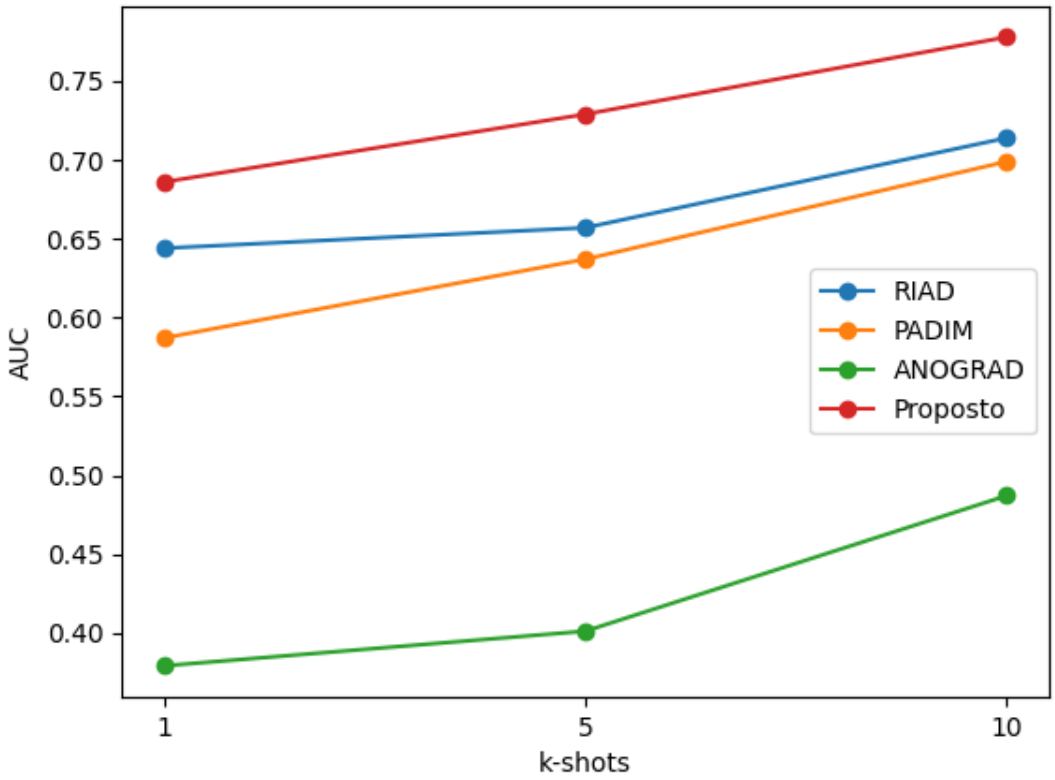
Como apresenta a Figura 29, onde apresenta um resumo dos resultados médios obtidos dos modelos nas configurações *1-shot*, *5-shot* e *10-shot*, a abordagem META-RIAD evidenciou resultados promissores chegando a superar os resultados médios em comparação

Tabela 11 – Resultados dos modelos para *10-shots* na métrica AUC-ROC (objetos e texturas)

Objeto	RIAD	PaDiM	AnoGrad	META-RIAD
bottle	0.971 ± 0.006	0.908 ± 0.011	0.649 ± 0.010	**0.982 ± 0.003**
cable	0.740 ± 0.011	0.698 ± 0.005	0.378 ± 0.032	**0.830 ± 0.007**
capsule	0.531 ± 0.015	**0.606 ± 0.004**	0.563 ± 0.013	0.540 ± 0.005
carpet	**0.750 ± 0.014**	0.470 ± 0.022	0.522 ± 0.036	0.663 ± 0.014
grid	0.714 ± 0.041	0.396 ± 0.024	0.460 ± 0.106	**0.781 ± 0.013**
hazelnut	0.697 ± 0.008	0.452 ± 0.011	0.517 ± 0.014	**0.720 ± 0.003**
leather	0.665 ± 0.012	0.521 ± 0.003	0.327 ± 0.057	**0.688 ± 0.010**
metal_nut	0.562 ± 0.018	0.377 ± 0.013	0.264 ± 0.058	**0.684 ± 0.023**
pill	0.549 ± 0.072	**0.704 ± 0.014**	0.487 ± 0.061	0.672 ± 0.002
screw	0.192 ± 0.011	**0.699 ± 0.025**	0.128 ± 0.036	0.575 ± 0.013
tile	0.834 ± 0.016	0.775 ± 0.005	0.405 ± 0.063	**0.853 ± 0.002**
toothbrush	0.824 ± 0.024	**0.960 ± 0.003**	0.525 ± 0.031	0.908 ± 0.004
transistor	0.690 ± 0.013	0.727 ± 0.015	0.705 ± 0.056	**0.834 ± 0.004**
wood	0.897 ± 0.029	0.808 ± 0.005	0.373 ± 0.034	**0.941 ± 0.003**
zipper	0.719 ± 0.010	0.729 ± 0.004	0.662 ± 0.019	**0.778 ± 0.004**

Fonte: O autor (2022)

Figura 29 – Resultados em AUC-ROC em relação aos K-shots



Fonte: O autor (2022)

aos modelos testados variando o K. Foi observado também que ao aumentar a quantidade de amostras oferecidas aos modelos, naturalmente há um crescimento no desempenho dos modelos.

Porém, em particular os resultados obtidos com o modelo AnoGrad (KWON et al., 2020)

foram bem inferiores aos demais, isso pode se dar devido a sua baixa expressividade do modelo proposto pelo autor.

Vale notar também que existe uma relação entre RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021) e a abordagem META-RIAD, pois como usa um modelo semelhante ao RIAD, que implementa uma arquitetura de uma rede U-Net, isso indica que a combinação da técnica MAML (FINN, 2017) com o modelo e técnicas similares ao RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021) implementadas da abordagem META-RIAD, pode aumentar a performance do mesmo, possibilitando que a técnica obtenha resultados satisfatórios com poucas amostras de treinamento.

5.3.2 Análise Qualitativa

A abordagem META-RIAD assim como RIAD e AnoGrad, podem ser consideradas como abordagens baseadas na reconstrução da imagem, uma vez que para produzir o índice de anomalia ou *anomaly score*, tais técnicas comparam a imagem de entrada com a imagem produzida pelo modelo, com isso tais modelos também podem ser classificados como modelos geradores.

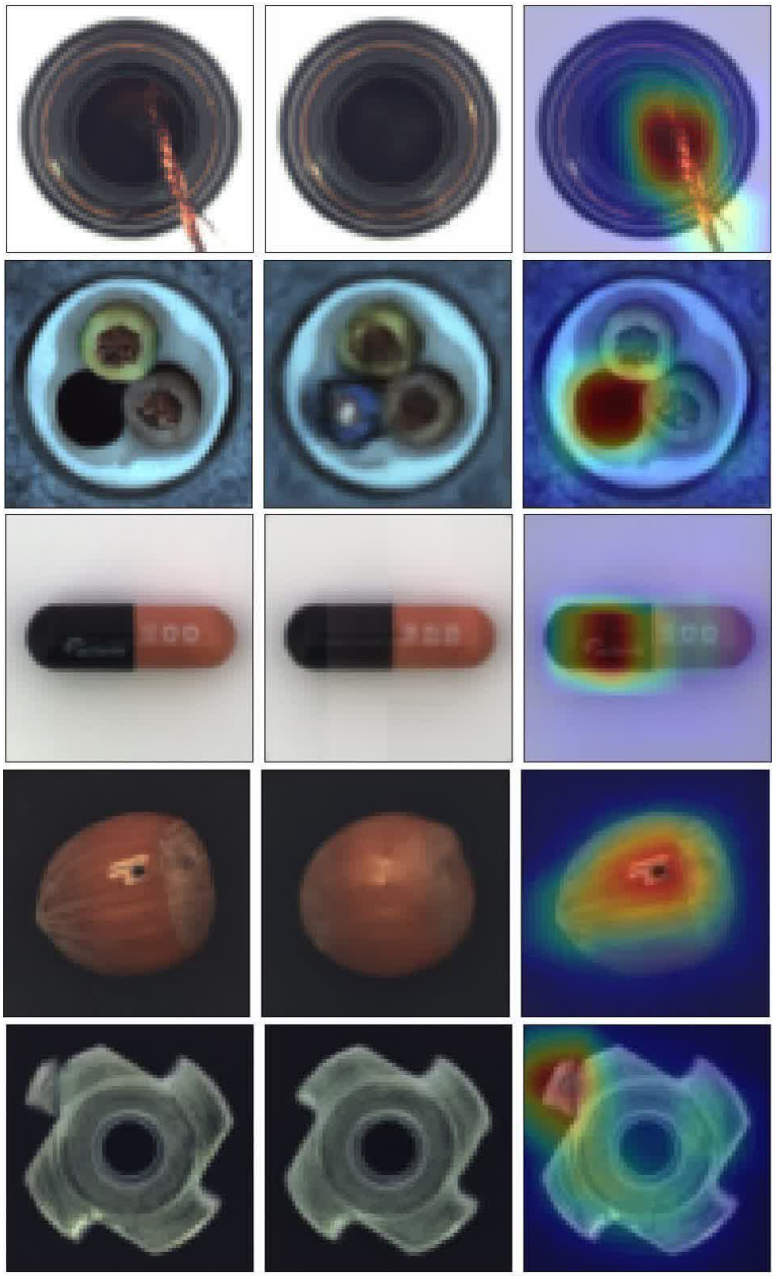
Assim, tanto a abordagem META-RIAD quanto a RIAD se baseiam na função de perda $MSGMS$, onde é uma variação em múltiplas escalas de GMS (XUE et al., 2013), tais produzem um índice indicando se no objeto da imagem contém uma anomalia e em sua maioria ao inspecionar essa saída do modelo em relação a função de perda $MSGMS$ conseguimos visualmente localizar a anomalia da imagem.

É valido observar visualmente tais imagens pois dá uma ideia de como o modelo esta realizando a classificação das imagens, com a indicação, correta ou incorreta, da existência ou não de anomalias. Caso o modelo esteja com o desempenho baixo, a imagem gerada pelo modelo bem como o mapa de calor produzido nos dá uma indicação de quais foram as características que causaram tal resultado. Além no mais, a visualização da localização da anomalia na imagem pode nos dar uma melhor interpretabilidade dos resultados, fornecendo informações de quais as regiões da imagem há um indício maior de anomalia.

Nos exemplos das Figuras 30 e 31 são apresentadas todas as imagens dos objetos que foram corretamente classificados como contendo alguma anomalia. Podemos observar, que o modelo apesar de somente 10 amostras serem usadas para a adaptação do modelo, o mesmo evidencia capacidade de estimar, através do mapa de calor produzido, a localização da anomalia na imagem dos objetos.

Na Figura 32, é visto mais uma vez o modelo indicando a localização da anomalia nas imagens de texturas. Apesar de em alguns exemplos a dispersão do mapa de calor ser maior, o modelo consegue aproximar a localização da anomalia com um índice de anomalia acentuado na região onde a anomalia na imagem da textura é apresentada.

Figura 30 – Resultados qualitativos da saída do modelo para os objetos com abordagem META-RIAD, capturada após 10 − *shots*. A coluna da esquerda representa a imagem de entrada, a do meio a imagem produzida pelo modelo e a terceira coluna é o mapa de calor representando a localização da anomalia

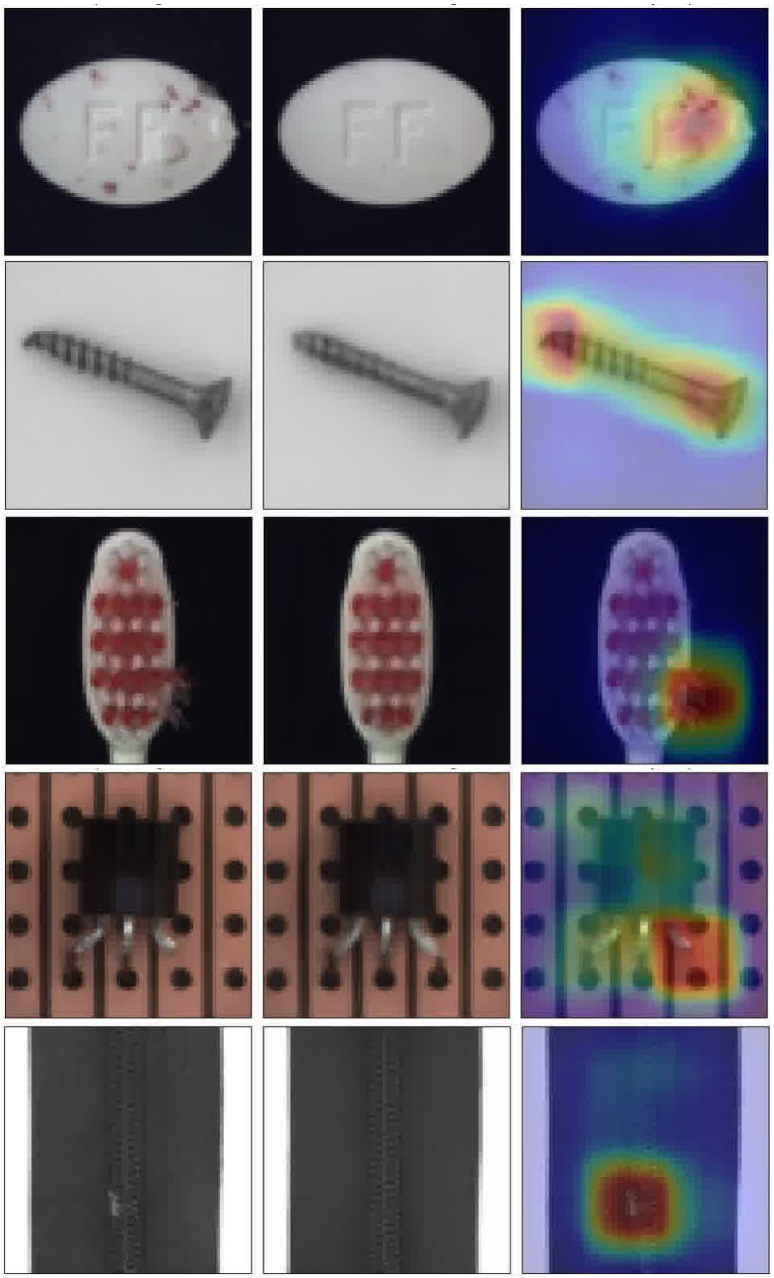


Fonte: O autor (2022)

Para comparar qualitativamente o modelo proposto, como mencionado anteriormente, o modelo RIAD também gera uma imagem como saída, sendo possível a visualização do seu mapa de calor com a indicação da região onde a anomalia se encontra na imagem.

Como podemos ver nas Figuras 33 e 34 são apresentadas as imagens de objetos que foram corretamente classificados como contendo alguma anomalia. Como visto, o modelo conseguiu gerar uma imagem do objeto em questão porém sem muitos detalhes e em alguns casos com muitos artefatos quadrados, isso pode estar sendo causado pelas máscaras

Figura 31 – Resultados qualitativos da saída do modelo para os objetos com abordagem META-RIAD, capturada após 10 – *shots*; Imagem apresenta a continuação da Figura 30

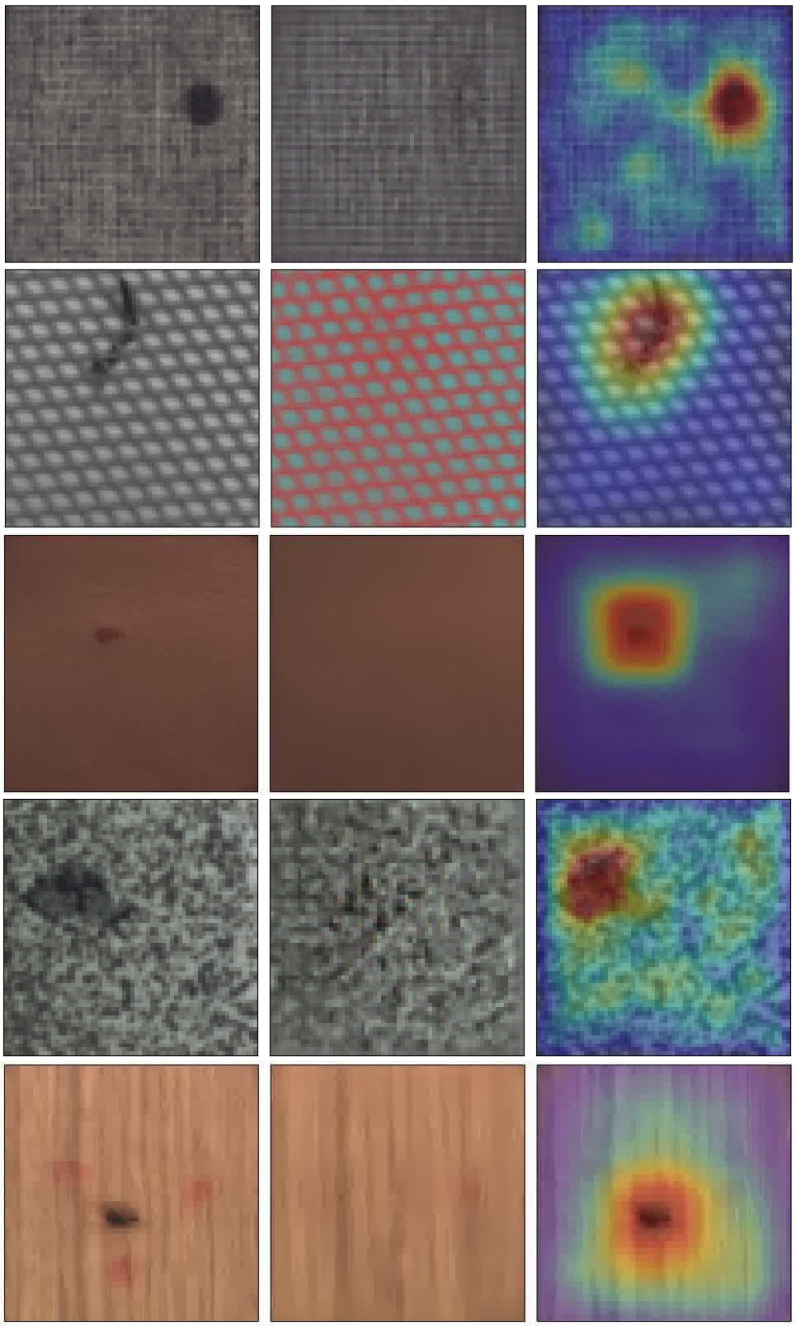


Fonte: O autor (2022)

usadas para realizar o procedimento de oclusão na imagem. O modelo RIAD usa a técnica de auto-pintura ou mais conhecida como *inpainting*, onde parte da imagem é removida e posteriormente preenchida com a imagem gerada pelo modelo. De maneira geral o modelo aprende a preencher essas lacunas na imagem que foram removidas com o objetivo de gerar uma imagem com a aparência e distribuição dos dados na qual o modelo foi treinado.

Da mesma forma na Figura 35, o modelo RIAD falha em alguns casos ao gerar imagens boas refletindo a distribuição dos dados aprendidos, esse comportamento indica que os dados informados no treinamento não foram suficientes para que o modelo adquirisse o

Figura 32 – Resultados qualitativos da saída do modelo para as texturas com abordagem META-RIAD, capturada após o treinamento da configuração 10 − *shots*. A coluna da esquerda representa a imagem de entrada, a do meio a imagem produzida pelo modelo e a direita é o mapa de calor representando a localização da anomalia

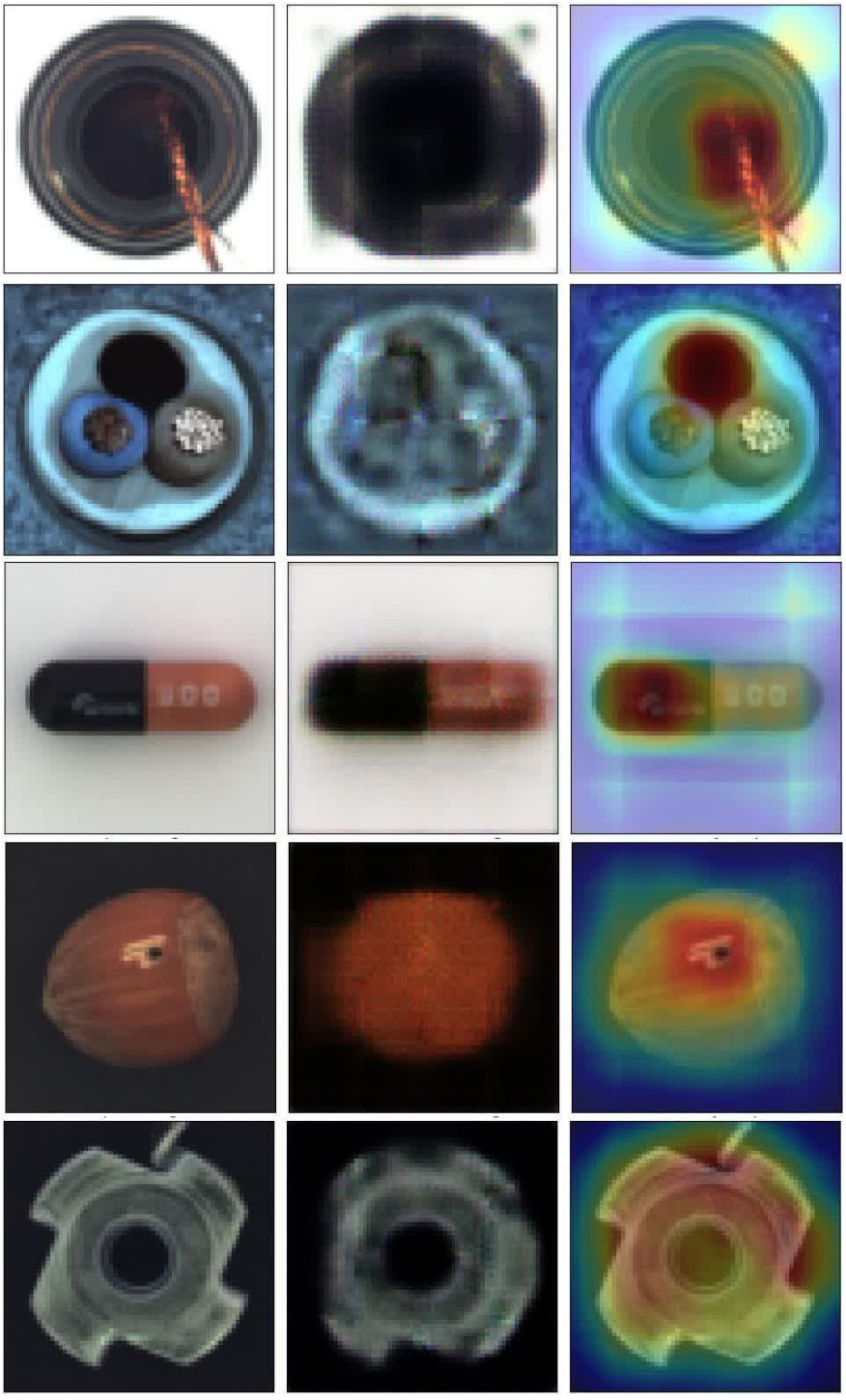


Fonte: O autor (2022)

conhecimento da distribuição dos dados, acarretando em uma performance inferior em relação ao modelo proposto.

O modelo AnoGrad, como mencionado acima, também baseia-se na reconstrução da imagem para a construção do índice de anomalia, entretanto o modelo não produz um mapa de calor para estimar a localização da anomalia da imagem, contudo, inspecionar a saída do modelo pode indicar o desempenho inferior nos experimentos realizados, uma

Figura 33 – Resultados qualitativos da saída do modelo para os objetos com RIAD, capturada após $10 - shots$. A coluna da esquerda representa a imagem de entrada, a do meio a imagem produzida pelo modelo e a terceira coluna é o mapa de calor representando a localização da anomalia

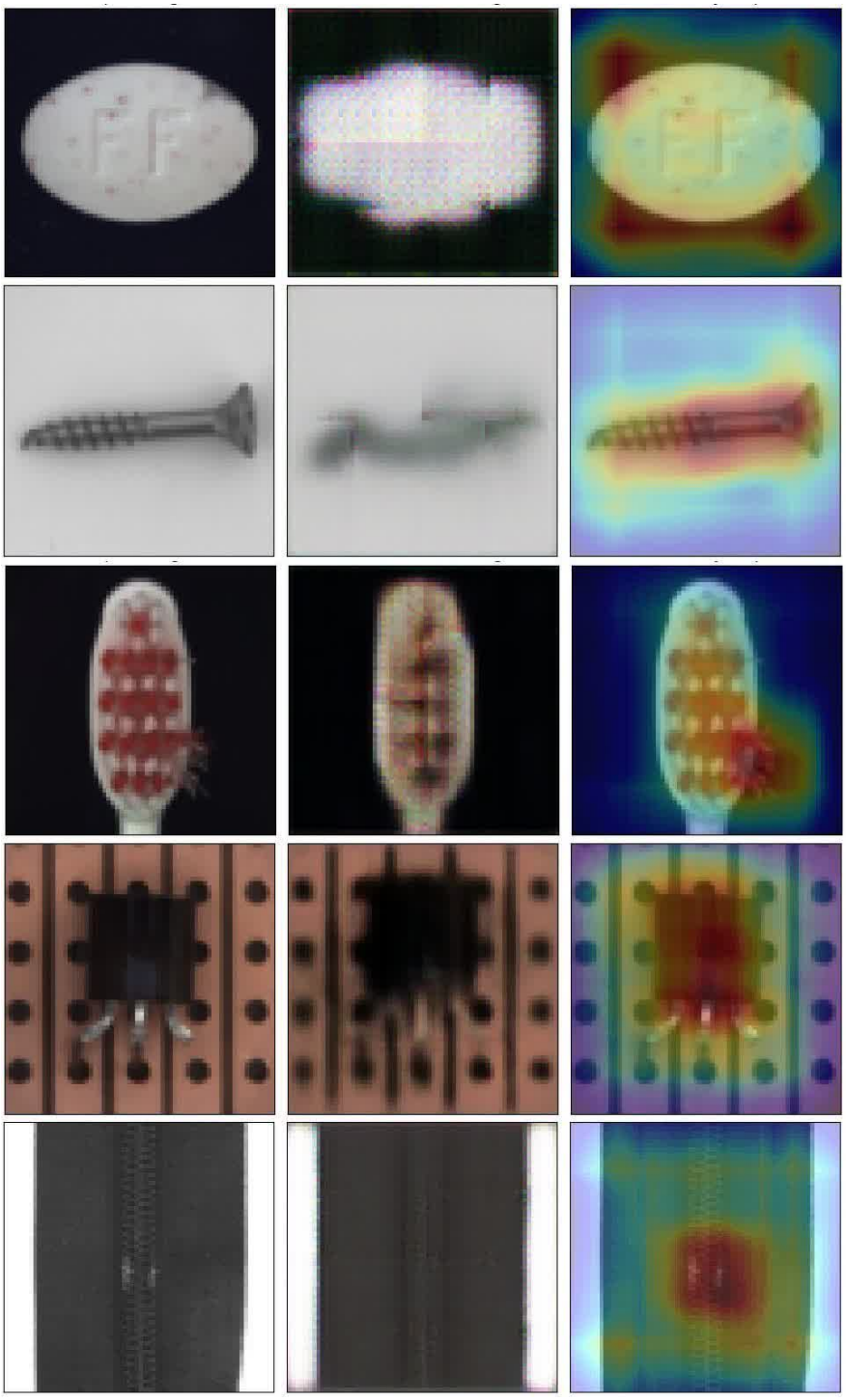


Fonte: O autor (2022)

vez que a qualidade dessas imagens geradas pelo modelo está diretamente relacionada a performance do mesmo em estimar se na imagem existe ou não uma anomalia.

Como podemos ver nas Figuras 36 e na Figura 37, o modelo AnoGrad não foi capaz

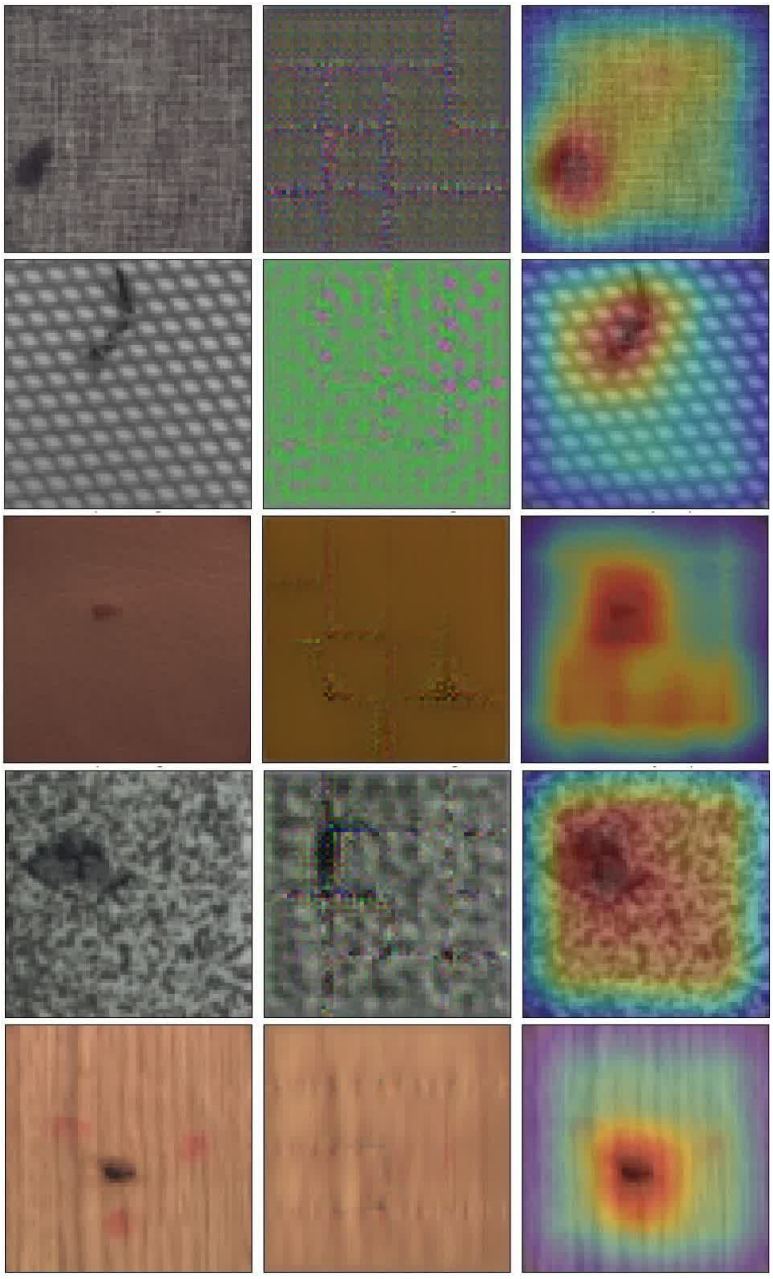
Figura 34 – Resultados qualitativos da saída do modelo para os objetos com RIAD, capturada após 10 – *shots*; Imagem apresenta a continuação da Figura 33



Fonte: O autor (2022)

de gerar imagens de objetos e texturas suficiente boas, ou seja, é esperado que o modelo gere uma imagem onde ao realizar a comparação com a imagem de entrada, seja capaz de produzir um erro característico da anomalia, entretanto aqui não é o caso, como todas as imagens estão muito discrepantes daquelas alimentadas na entrada do modelo, o erro sempre será grande para todos os casos tanto para as imagens com anomalia quanto

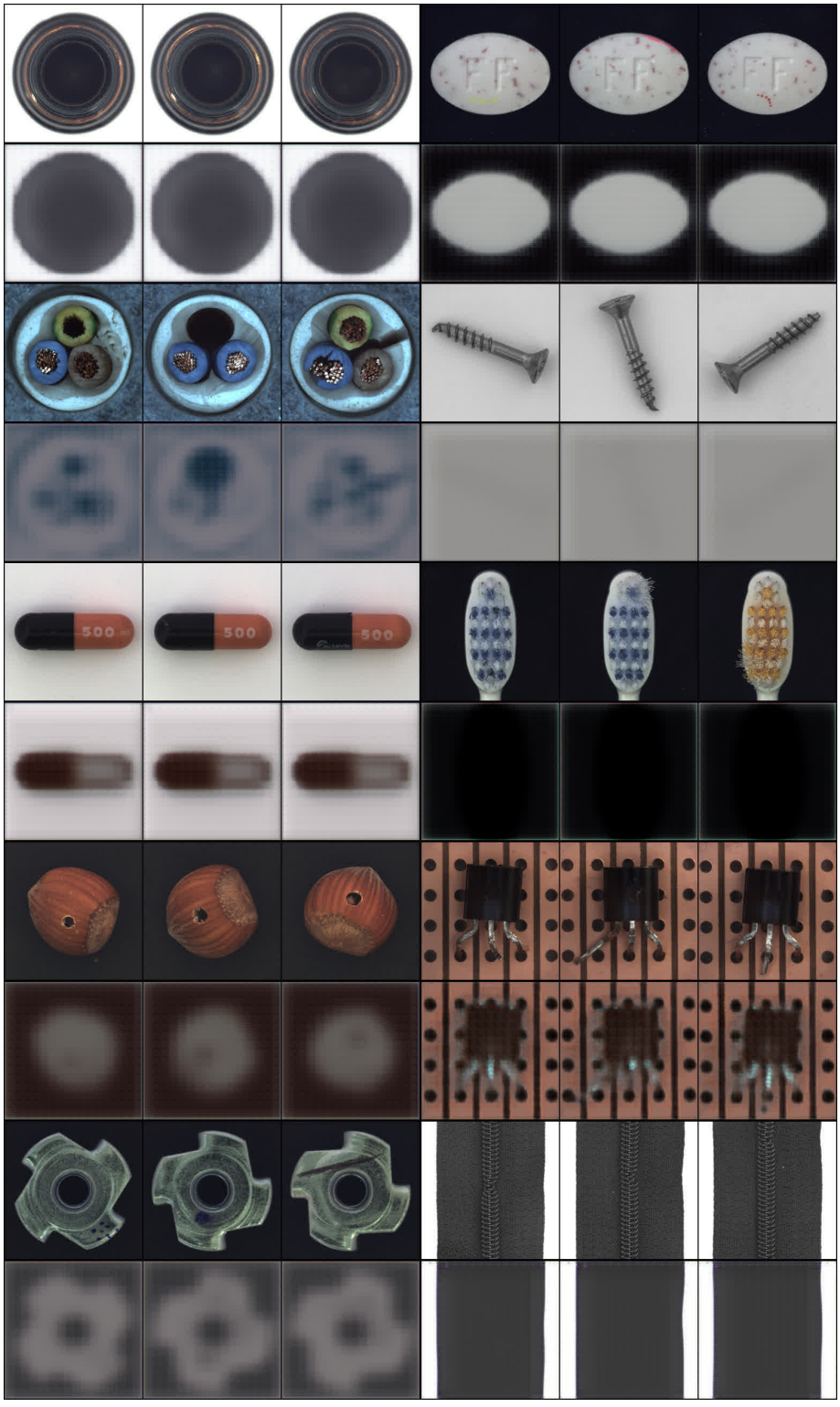
Figura 35 – Resultados qualitativos da saída do modelo para as texturas com RIAD, capturada após o treinamento da configuração $10 - shots$. A coluna da esquerda representa a imagem de entrada, a do meio a imagem produzida pelo modelo e a direita é o mapa de calor representando a localização da anomalia.



Fonte: O autor (2022)

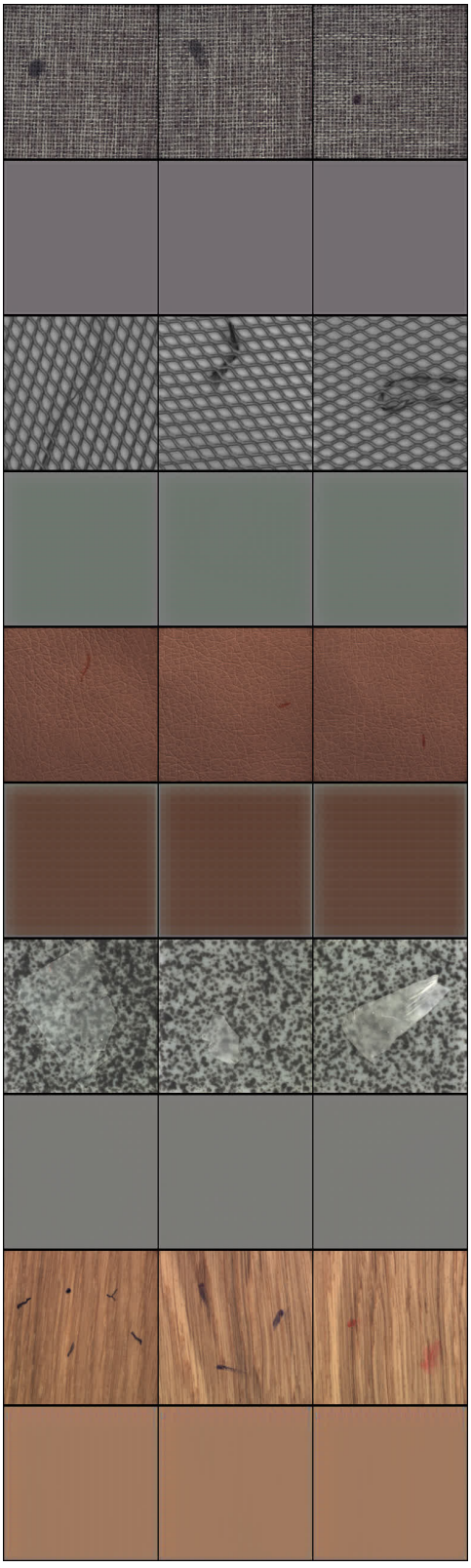
aquelas livres de anomalia, fazendo assim com o que o modelo tenha um desempenho muito inferior em relação aos demais expostos a mesma configuração de treinamento $10 - shots$.

Figura 36 – Amostras das imagens geradas pelo modelo AnoGrad após o treinamento da configuração 10 − *shots* em relação a cada objeto. Nas linhas ímpares amostras de imagens de entrada, nas pares a imagem gerada pelo modelo.



Fonte: O autor (2022)

Figura 37 – Amostras das imagens geradas pelo modelo AnoGrad após o treinamento da configuração 10 − *shots* em relação a cada textura. Nas linhas ímpares amostras de imagens de entrada, nas pares a imagem gerada pelo modelo.



Fonte: O autor (2022)

6 CONCLUSÃO

6.1 CONSIDERAÇÕES FINAIS

Esse trabalho, teve como objetivo apresentar uma nova abordagem para detecção de anomalias em imagens que, assumindo a premissa de que somente amostras ditas como normais estão contidas em seu processo de treinamento, a abordagem seja capaz de obter um bom desempenho com poucos exemplos nunca vistos no conjunto de treinamento, através do uso de algoritmos de meta-aprendizado.

Para que o objetivo fosse atingido, foi desenvolvida uma abordagem baseada em meta-aprendizado para a detecção de anomalias em imagens necessitando apenas poucas amostras de imagens livres de anomalias. Foi definido um protocolo de avaliação onde os modelos e a abordagem, similarmente a (LU et al., 2020), os modelos foram pré-treinados para a comparação a abordagem do treinado com MAML (FINN, 2017). A abordagem META-RIAD e os modelos foram comparados quantitativamente e qualitativamente, sendo que a abordagem META-RIAD demonstrou resultados promissores.

Podemos notar que a abordagem META-RIAD teve destaque nas configurações de avaliação *1-shot*, *5-shots* e *10-shots*, demonstrando também que ao aumentar a quantidade de amostras na avaliação dos modelos, o desempenho dos mesmos melhora proporcionalmente. Na avaliação qualitativa, ficou demonstrado que a abordagem META-RIAD apresentou resultados promissores no quesito da localização das anomalias nas imagens testadas. Em particular o modelo RIAD, apresentou resultados qualitativamente mais próximos da abordagem META-RIAD, porém ainda com deficiência em alguns cenários de localização das anomalias. Apesar da relação que há entre a abordagem META-RIAD e o modelo RIAD, ficou evidente que o algoritmo MAML pode melhorar o desempenho do modelo com relação à eficiência dos dados para um desempenho aceitável do mesmo.

A abordagem META-RIAD usou como modelo uma Rede Neural U-Net, sendo assim tal abordagem pode ser adaptada para outros cenários, como exemplo a detecção de anomalias em vídeos.

6.2 CONTRIBUIÇÕES

A principal contribuição deste trabalho foi o desenvolvimento de uma combinação entre as técnicas do RIAD (ZAVRTANIK; KRISTAN; SKOČAJ, 2021) como modelo adotado juntamente com o processo de *inpating* e o algoritmo de meta-aprendizado MAML (FINN, 2017) para a detecção e localização de anomalias em imagens de objetos e texturas. Também foi proposto uma modificação do algoritmo de mascaramento ou remoção de pixeis no processo de *inpainting* do modelo RIAD. Além disso o algoritmo MAML demonstrou

que pode ser flexível para diferentes modelos e processos de aprendizagem de máquina, apesar de adicionar ser demorado ao realizar o processo de otimização.

Foi apresentada uma comparação da abordagem META-RIAD com modelos considerados estado da arte, ao menos aqueles que no momento da realização dos experimentos deste trabalho eram considerados os que se destacavam nas bases de dados comumente utilizadas para a tarefa de detecção de anomalias.

Outra contribuição desde trabalho foi uma nova forma de avaliar modelos de detecção de anomalias em imagens com um conjunto restrito de amostras na base MVTec-AD (BERGMANN et al., 2019). Tal protocolo de avaliação pode ser estendido para a avaliação de mais de uma categoria avaliada por vez e entre outras configurações de detecção de anomalias em *K-Shots*.

Além disso, este trabalho demonstra uma visão de como abordar o problema de detecção de anomalias em imagens, onde o número de amostras é limitada e usualmente estão disponíveis somente amostras do objeto em questão considerado sem defeito ou sem anomalias, tornado assim a abordagem META-RIAD compatível com um cenário real de um sistema de detecção de anomalias, onde o modelo consegue se adaptar com poucos exemplos para o contexto no qual o modelo será empregado.

6.3 LIMITAÇÕES E AMEAÇAS À VALIDADE

A abordagem META-RIAD tem como premissa que a base de dados inicial para o treinamento deve ser de tamanho significativo porém não muito grande, como visto na Tabela 1, ou seja, o conjunto de $M - 1$ categorias.

Entretanto, para os novos objetos ou texturas que são adaptados na fase de Meta Teste, a abordagem MAML não requer uma quantidade de amostras grande, como evidenciado neste trabalho, portanto com somente uma amostra a abordagem consegue obter um bom desempenho.

Sendo assim, experimentos futuros podem validar se a técnica MAML consegue obter bons resultados com uma base inicial muito pequena, bem como se ao usar outras bases de dados, por exemplo ImageNet (DENG et al., 2009), como conjunto de treinamento, poderiam influenciar nos resultados mesmo sendo base de dados de contextos diferentes.

6.4 TRABALHOS FUTUROS

Neste trabalho o algoritmo MAML (FINN, 2017) foi empregado para realizar o processo de treinamento e adaptação do modelo, em seu processo ao propagar o gradiente para a atualização dos pesos, usa derivadas de segunda ordem, sendo assim tornando o treinamento muito lento e limitando o hardware usado para tal experimento. Uma alterna-

tiva é explorar ou desenvolver uma solução que ataque esse problema, tais como Reptile (NICHOL; ACHIAM; SCHULMAN, 2018), onde sugere considerar somente as derivadas de primeira ordem, ou iMAML (RAJESWARAN et al., 2019) no qual atribui uma regularização entre θ e θ', removendo parte do custo computacional ao realizar as atualizações dos pesos.

Explorar outros modelos além do U-Net (RONNEBERGER, 2015) adotado, pesquisas recentes apontam que Vision Transformer (ViT) (DOSOVITSKIY et al., 2021) tem demonstrado resultados promissores, sendo assim pode-se investigar como esse ViT pode melhorar os resultados e sua aplicabilidade no MAML (FINN, 2017) ou outro algoritmo similar.

A base utilizada nos experimentos contém objetos das mais diversas características, porém compartilham um cenário mais controlado e restrito. Sendo assim pode-se investigar a performance desse algoritmo com bases mais complexas como bases de raio-x do tórax ou uma mistura de bases de raio-x de diversas partes do corpo.

Como foi evidenciado neste trabalho, somente uma categoria é avaliada na detecção de anomalia por vez, ou seja, o modelo é treinado com $M - 1$ categorias, após isso o modelo é adaptado e avaliado na categoria faltante. Entretanto, é interessante explorar a capacidade da abordagem META-RIAD de reter o desempenho de identificar anomalias uma determinada categoria após a adaptação de uma outra categoria. Essa exploração está relacionada ao problema conhecido como *Catastrophic Forgetting*, onde o modelo esquece completamente e abruptamente as informações aprendidas anteriormente ao aprender novas informações, sendo um problema antigo (MCCLOSKEY; COHEN, 1989) e com várias pesquisas recentes que tratam desse problema (KEMKER et al., 2018), (HU et al., 2018), (ERGüN; TöREYIN, 2020), (KE et al., 2020). Assim, o problema pode ser visto também no contexto de *Continual Learning* ou *Lifelong Learninig*, tal paradigma de aprendizagem aprende continuamente acumulando conhecimento passado. Bem como explorar e analisar o problema de *Curriculum Learning*, onde o modelo é avaliado começando com apenas exemplos considerados fáceis e depois aumenta gradualmente a dificuldade das tarefas.

Além disso, diante desses trabalhos futuros, a realização de uma publicação deste trabalho, formato de artigo, em uma conferência ou periódico científico da área.

REFERÊNCIAS

ASSYLBEKOV, Z.; MELNYKOV, I.; BEKISHEV, R.; BALTABAYEVA, A.; BISSENGALIYEVA, D.; MAMLIN, E. Detecting value-added tax evasion by business entities of kazakhstan. In: SPRINGER. *International Conference on Intelligent Decision Technologies*. [S.l.], 2016. p. 37–49.

BAUR, C.; WIESTLER, B.; ALBARQOUNI, S.; NAVAB, N. Deep autoencoding models for unsupervised anomaly segmentation in brain mr images. In: SPRINGER. *International MICCAI Brainlesion Workshop*. [S.l.], 2018. p. 161–169.

BERGMANN, P.; FAUSER, M.; SATTLEGGER, D.; STEGER, C. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2019. p. 9592–9600.

BERGMANN, P.; LÖWE, S.; FAUSER, M.; SATTLEGGER, D.; STEGER, C. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *arXiv preprint arXiv:1807.02011*, 2018.

BERTALMIO, M.; SAPIRO, G.; CASELLES, V.; BALLESTER, C. Image inpainting. In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. [S.l.: s.n.], 2000. p. 417–424.

BROMLEY, J.; BENTZ, J. W.; BOTTOU, L.; GUYON, I.; LECUN, Y.; MOORE, C.; SÄCKINGER, E.; SHAH, R. Signature verification using a "siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, World Scientific, v. 7, n. 04, p. 669–688, 1993.

CHALAPATHY, R.; CHAWLA, S. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*, 2019.

CHANDOLA; VARUN; BANERJEE; ARINDAM; KUMAR; VIPIN. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, ACM, v. 41, n. 3, p. 15, 2009.

CHAUDHARY; AMITABH; SZALAY; S, A.; W, M. A. Very fast outlier detection in large multidimensional data sets. In: *DMKD*. [S.l.: s.n.], 2002.

CHEN, K.; LU, S.; TENG, H. Adaptive real-time anomaly detection using inductively generated sequential patterns,". In: *Fifth Intrusion Detection Workshop, SRI International, Menlo Park, CA*. [S.l.: s.n.], 1990.

CORTES, C.; VAPNIK, V. Support-vector networks. *Machine learning*, Springer, v. 20, n. 3, p. 273–297, 1995.

DEFARD, T.; SETKOV, A.; LOESCH, A.; AUDIGIER, R. Padim: a patch distribution modeling framework for anomaly detection and localization. *arXiv preprint arXiv:2011.08785*, 2020.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In: IEEE. *2009 IEEE conference on computer vision and pattern recognition*. [S.l.], 2009. p. 248–255.

DENG, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, IEEE, v. 29, n. 6, p. 141–142, 2012.

DESFORGES, M.; JACOB, P.; COOPER, J. Applications of probability density estimation to the detection of abnormal conditions in engineering. *Proceedings of the institution of mechanical engineers, part c: Journal of mechanical engineering science*, SAGE Publications Sage UK: London, England, v. 212, n. 8, p. 687–703, 1998.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S.; USZKOREIT, J.; HOULSBY, N. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

ERGüN, E.; TöREYIN, B. U. Continual learning with sparse progressive neural networks. In: *2020 28th Signal Processing and Communications Applications Conference (SIU)*. [S.l.: s.n.], 2020. p. 1–4.

FINN, e. a. Model-agnostic meta-learning for fast adaptation of deep networks. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2017. p. 1126–1135.

FUJIMAKI, R.; YAIRI, T.; MACHIDA, K. An approach to spacecraft anomaly detection problem using kernel feature space. In: *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. [S.l.: s.n.], 2005. p. 401–410.

GONG, D.; LIU, L.; LE, V.; SAHA, B.; MANSOUR, M. R.; VENKATESH, S.; HENGEL, A. v. d. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2019.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

GOODFELLOW, I.; POUGET-ABADIE, J.; MIRZA, M.; XU, B.; WARDE-FARLEY, D.; OZAIR, S.; COURVILLE, A.; BENGIO, Y. Generative adversarial nets. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2014. p. 2672–2680.

GRAVES, A.; WAYNE, G.; DANIHELKA, I. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HU, W.; LIN, Z.; LIU, B.; TAO, C.; TAO, Z.; MA, J.; ZHAO, D.; YAN, R. Overcoming catastrophic forgetting for continual learning via model adaptation. In: *International conference on learning representations*. [S.l.: s.n.], 2018.

HUSH, D. R.; HORNE, B. G. Progress in supervised neural networks. *IEEE signal processing magazine*, IEEE, v. 10, n. 1, p. 8–39, 1993.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: PMLR. *International conference on machine learning*. [S.l.], 2015. p. 448–456.

JEONG, T.; KIM, H. Ood-maml: Meta-learning for few-shot out-of-distribution detection and classification. *Advances in Neural Information Processing Systems*, v. 33, 2020.

KE, Z.; LIU, B.; WANG, H.; SHU, L. Continual learning with knowledge transfer for sentiment classification. In: SPRINGER. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. [S.l.], 2020. p. 683–698.

KEMKER, R.; MCCLURE, M.; ABITINO, A.; HAYES, T.; KANAN, C. Measuring catastrophic forgetting in neural networks. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2018. v. 32, n. 1.

KOCH, G.; ZEMEL, R.; SALAKHUTDINOV, R. et al. Siamese neural networks for one-shot image recognition. In: LILLE. *ICML deep learning workshop*. [S.l.], 2015. v. 2.

KRIZHEVSKY, A.; NAIR, V.; HINTON, G. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, v. 55, n. 5, 2014.

KRIZHEVSKY, A. et al. Learning multiple layers of features from tiny images. Citeseer, 2009.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2012. p. 1097–1105.

KWON, G.; PRABHUSHANKAR, M.; TEMEL, D.; ALREGIB, G. Backpropagated gradient representations for anomaly detection. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. [S.l.: s.n.], 2020.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Ieee, v. 86, n. 11, p. 2278–2324, 1998.

LOGOTHETIS, N. K. Vision: a window on consciousness. *Scientific American*, JSTOR, v. 281, n. 5, p. 68–75, 1999.

LU, Y.; YU, F.; REDDY, M. K. K.; WANG, Y. Few-shot scene-adaptive anomaly detection. In: SPRINGER. *European Conference on Computer Vision*. [S.l.], 2020. p. 125–141.

MACÊDO, D.; REN, T. I.; ZANCHETTIN, C.; OLIVEIRA, A. L.; LUDERMIR, T. Entropic out-of-distribution detection: Seamless detection of unknown examples. *arXiv preprint arXiv:2006.04005*, 2020.

MAHONEY, M. V.; CHAN, P. K. Learning nonstationary models of normal network traffic for detecting novel attacks. In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2002. p. 376–385.

MCCLOSKEY, M.; COHEN, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In: BOWER, G. H. (Ed.). Academic Press, 1989, (Psychology of Learning and Motivation, v. 24). p. 109–165. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0079742108605368>.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, v. 5, n. 4, p. 115–133, Dec 1943. ISSN 1522-9602. Disponível em: <https://doi.org/10.1007/BF02478259>.

MCLACHLAN, G. J. Mahalanobis distance. *Resonance*, Springer, v. 4, n. 6, p. 20–26, 1999.

METZ, L.; POOLE, B.; PFAU, D.; SOHL-DICKSTEIN, J. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: *Icml*. [S.l.: s.n.], 2010.

NETZER, Y.; WANG, T.; COATES, A.; BISSACCO, A.; WU, B.; NG, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.

NICHOL, A.; ACHIAM, J.; SCHULMAN, J. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

NIELSEN, M. A. *Neural networks and deep learning*. [S.l.]: Determination press San Francisco, CA, USA, 2015. v. 25.

OLAH, C.; CAMMARATA, N.; SCHUBERT, L.; GOH, G.; PETROV, M.; CARTER, S. An overview of early vision in inceptionv1. *Distill*, 2020. Https://distill.pub/2020/circuits/early-vision.

PANG, G.; SHEN, C.; CAO, L.; HENGEL, A. V. D. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, ACM New York, NY, USA, v. 54, n. 2, p. 1–38, 2021.

PARK, H.; NOH, J.; HAM, B. Learning memory-guided normality for anomaly detection. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2020.

QIN, M.; HWANG, K. Frequent episode rules for intrusive anomaly detection with internet datamining. In: CITESEER. *USENIX Security Symposium*. [S.l.], 2004. p. 1–15.

RAGHU, M.; POOLE, B.; KLEINBERG, J.; GANGULI, S.; SOHL-DICKSTEIN, J. On the expressive power of deep neural networks. In: PMLR. *international conference on machine learning*. [S.l.], 2017. p. 2847–2854.

RAJESWARAN, A.; FINN, C.; KAKADE, S. M.; LEVINE, S. Meta-learning with implicit gradients. *Advances in Neural Information Processing Systems*, v. 32, p. 113–124, 2019.

RAJPURKAR, P.; IRVIN, J.; ZHU, K.; YANG, B.; MEHTA, H.; DUAN, T.; DING, D.; BAGUL, A.; LANGLOTZ, C.; SHPANSKAYA, K. et al. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint arXiv:1711.05225*, 2017.

RAMADAS; MANIKANTAN; OSTERMANN; SHAWN; TJADEN; BRETT. Detecting anomalous network traffic with self-organizing maps. In: SPRINGER. *International Workshop on Recent Advances in Intrusion Detection*. [S.l.], 2003. p. 36–54.

RIESENHUBER, M.; POGGIO, T. Hierarchical models of object recognition in cortex. *Nature neuroscience*, Nature Publishing Group, v. 2, n. 11, p. 1019–1025, 1999.

RONNEBERGER, e. a. U-net: Convolutional networks for biomedical image segmentation. In: SPRINGER. *International Conference on Medical image computing and computer-assisted intervention*. [S.l.], 2015. p. 234–241.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958.

RUDER, S. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

RUFF, L.; VANDERMEULEN, R.; GOERNITZ, N.; DEECKE, L.; SIDDIQUI, S. A.; BINDER, A.; MÜLLER, E.; KLOFT, M. Deep one-class classification. In: PMLR. *International conference on machine learning*. [S.l.], 2018. p. 4393–4402.

SANTORO, e. a. Meta-learning with memory-augmented neural networks. In: PMLR. *International conference on machine learning*. [S.l.], 2016. p. 1842–1850.

SCHLEGL, T.; SEEBÖCK, P.; WALDSTEIN, S. M.; SCHMIDT-ERFURTH, U.; LANGS, G. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In: SPRINGER. *International conference on information processing in medical imaging*. [S.l.], 2017. p. 146–157.

SCHÖLKOPF, B.; PLATT, J. C.; SHAWE-TAYLOR, J.; SMOLA, A. J.; WILLIAMSON, R. C. Estimating the support of a high-dimensional distribution. *Neural computation*, MIT Press, v. 13, n. 7, p. 1443–1471, 2001.

SEHWAG, V.; CHIANG, M.; MITTAL, P. Ssd: A unified framework for self-supervised outlier detection. In: *International Conference on Learning Representations*. [s.n.], 2021. Disponível em: <https://openreview.net/forum?id=v5gjXpmR8J>.

SERRE, T. Hierarchical models of the visual system. In: _____. *Encyclopedia of Computational Neuroscience*. New York, NY: Springer New York, 2013. p. 1–12. ISBN 978-1-4614-7320-6. Disponível em: <https://doi.org/10.1007/978-1-4614-7320-6_345-1>.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

SZEGEDY, C.; LIU, W.; JIA, Y.; SERMANET, P.; REED, S.; ANGUELOV, D.; ERHAN, D.; VANHOUCKE, V.; RABINOVICH, A. Going deeper with convolutions. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 1–9.

TACK, J.; MO, S.; JEONG, J.; SHIN, J. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2020.

TANG, Y.-X.; TANG, Y.-B.; HAN, M.; XIAO, J.; SUMMERS, R. M. Abnormal chest x-ray identification with generative adversarial one-class classifier. In: IEEE. *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. [S.l.], 2019. p. 1358–1361.

TARASSENKO, L.; HAYTON, P.; CERNEAZ, N.; BRADY, M. Novelty detection for the identification of masses in mammograms. IET, 1995.

TAX, D. M.; DUIN, R. P. Support vector data description. *Machine learning*, Springer, v. 54, n. 1, p. 45–66, 2004.

VINYALS, O.; BLUNDELL, C.; LILLICRAP, T.; WIERSTRA, D. et al. Matching networks for one shot learning. *Advances in neural information processing systems*, v. 29, p. 3630–3638, 2016.

WANG, Z.; BOVIK, A. C.; SHEIKH, H. R.; SIMONCELLI, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, IEEE, v. 13, n. 4, p. 600–612, 2004.

WENG, L. Meta-learning: Learning to learn fast. *lilianweng.github.io/lil-log*, 2018. Disponível em: <http://lilianweng.github.io/lil-log/2018/11/29/meta-learning.html>.

WENG, L. Meta reinforcement learning. *lilianweng.github.io/lil-log*, 2019. Disponível em: <http://lilianweng.github.io/lil-log/2019/06/23/meta-reinforcement-learning.html>.

XU, K.; BA, J.; KIROS, R.; CHO, K.; COURVILLE, A.; SALAKHUDINOV, R.; ZEMEL, R.; BENGIO, Y. Show, attend and tell: Neural image caption generation with visual attention. In: PMLR. *International conference on machine learning*. [S.l.], 2015. p. 2048–2057.

XUE, W.; ZHANG, L.; MOU, X.; BOVIK, A. C. Gradient magnitude similarity deviation: A highly efficient perceptual image quality index. *IEEE Transactions on Image Processing*, IEEE, v. 23, n. 2, p. 684–695, 2013.

YI, J.; YOON, S. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In: *Proceedings of the Asian Conference on Computer Vision*. [S.l.: s.n.], 2020.

YPMA; ALEXANDER; DUIN; PW, R. Novelty detection using self-organizing maps. In: CITESEER. *ICONIP (2)*. [S.l.], 1997. p. 1322–1325.

YU; DANTONG; SHEIKHOLESLAMI; GHOLAMHOSEIN; ZHANG; AIDONG. Findout: Finding outliers in very large datasets. *Knowledge and information Systems*, Springer, v. 4, n. 4, p. 387–412, 2002.

ZAVRTANIK, V.; KRISTAN, M.; SKOČAJ, D. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition*, Elsevier, v. 112, p. 107706, 2021.

ZHANG, H.; XU, T.; LI, H.; ZHANG, S.; WANG, X.; HUANG, X.; METAXAS, D. N. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In: *Proceedings of the IEEE international conference on computer vision*. [S.l.: s.n.], 2017. p. 5907–5915.

ZHANG, S.; YE, F.; WANG, B.; HABETLER, T. G. Few-shot bearing anomaly detection via model-agnostic meta-learning. In: IEEE. *2020 23rd International Conference on Electrical Machines and Systems (ICEMS)*. [S.l.], 2020. p. 1341–1346.

ZHOU, Y.; LIANG, X.; ZHANG, W.; ZHANG, L.; SONG, X. Vae-based deep svdd for anomaly detection. *Neurocomputing*, Elsevier, v. 453, p. 131–140, 2021.