UNIVERSIDADE FEDERAL DO PARANÁ

DANIEL PIMENTA FURTADO

CLASSIFICAÇÃO DE CORAIS UTILIZANDO APRENDIZAGEM PROFUNDA

CURITIBA PR

2022

DANIEL PIMENTA FURTADO

CLASSIFICAÇÃO DE CORAIS UTILIZANDO APRENDIZAGEM PROFUNDA

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Informática no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Luiz Eduardo S. Oliveira.

Coorientador: Marco A. Zanata Alves.

CURITIBA PR

2022

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA CIÊNCIA E TECNOLOGIA

Furtado, Daniel Pimenta
Classificação de corais utilizando aprendizagem profunda. / Daniel Pimenta Furtado. – Curitiba, 2022.
1 recurso on-line : PDF.

Dissertação (Mestrado) - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática.

Orientador: Luiz Eduardo S. Oliveira.
Coorientador: Marco A. Zanata Alves

1. Redes Neurais (Computação). 2. Aprendizado do computador. 3. Ecologia marinha. I. Oliveira, Luiz Eduardo S. II. Alves, Marco A. Zanata. III. Universidade Federal do Paraná. Programa de Pós-Graduação em Informática. IV. Título.

Bibliotecária: Roseny Rivelini Morciani CRB-9/1585

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da Dissertação de Mestrado de **DANIEL PIMENTA FURTADO** intitulada: **Classificação de corais utilizando Aprendizagem Profunda**, sob orientação do Prof. Dr. LUIZ EDUARDO SOARES DE OLIVEIRA, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de mestre está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 03 de Outubro de 2022.

Assinatura Eletrônica
04/10/2022 13:29:08.0
LUIZ EDUARDO SOARES DE OLIVEIRA
Presidente da Banca Examinadora

Assinatura Eletrônica
16/10/2022 16:16:56.0
GUILHERME ORTIGARA LONGO
Avaliador Externo (UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE)

Assinatura Eletrônica
04/10/2022 19:29:08.0
PAULO RICARDO LISBOA DE ALMEIDA
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

Aos meus pais, meus maiores incentivadores.

AGRADECIMENTOS

A Deus, porque d'Ele, por Ele e para Ele são todas as coisas.

Aos meus pais, pelas orações e encorajamento, por cada conselho, mesmo estando longe fisicamente.

A minha irmã Marcela e meu cunhado Marlos, por me receberem em sua casa durante a pandemia. Esse trabalho é só uma das histórias que construímos juntos.

Aos meus orientadores Prof. Luiz Eduardo S. Oliveira e Prof. Marco A. Zanata Alves, pelo ensino, sabedoria, paciência, compreensão, disponibilidade para que esse trabalho pudesse ser produzido.

Aos meus colegas do HiPES que trouxeram leveza e alegria em meio às dificuldades, demonstrando companheirismo durante todo o período do curso.

À Capes pelo auxílio financeiro que permitiu a boa realização desse trabalho.

Ao Instituto Serrapilheira, pela concessão de recursos financeiros proporcionados.

À UFPR, ao DINF e ao PPGInf, pela oportunidade concedida e estrutura proporcionada.

Aos meus amigos de Minas Gerais e aos meus novos amigos de Curitiba, por estarem presentes em todo esse processo orando e torcendo por mim.

RESUMO

Os corais são animais coloniais do Filo Cnidaria que formam os recifes de corais os quais desempenham um papel fundamental no ecossistema marinho, servindo de habitat para peixes, moluscos, crustáceos, esponjas, e diversos outros organismos. Entretanto, em função das mudanças climáticas e consequentemente do aumento da temperatura dos oceanos, alguns corais estão perdendo coloração ao perder as algas que vivem em simbiose com esses animais, podendo causar alta mortalidade. Devido à sua importância para o ambiente marinho, o monitoramento dos recifes de corais é essencial e realiza-se principalmente por meio de dados extraídos de fotografias subaquáticas. Nesse contexto o Laboratório de Ecologia Marinha (LECOM) da Universidade Federal do Rio Grande do Norte (UFRN) desenvolveu o projeto "#DeOlhoNosCorais" o qual incentiva pessoas a postarem fotos de recifes de corais em suas redes sociais. Desse modo, o laboratório obtém informações da saúde dos corais na costa brasileira. Todavia, a análise dessas imagens é atualmente realizada manualmente e com a necessidade de um especialista. Nessa dissertação trabalhamos em parceria com o LECOM para efetuar o treinamento de modelos de aprendizagem de máquina para realizar a classificação, segmentação e localização de objetos das postagens em redes sociais com a *hashtag* "#DeOlhoNosCorais", reduzindo assim o tempo empregado na análise das imagens. Um diferencial da base de dados utilizada no trabalho em relação às presentes na literatura é a presença do mapa de segmentação que é a classificação de cada píxel contido na imagem. Portanto, avaliou-se os modelos de aprendizagem de máquina frente a 3 cenários: classificação, segmentação semântica e localização de objetos. Também avaliou-se o uso de *transfer learning* utilizando a base de dados *Pacific Labeled Corals* (PLC) e utilizou-se o algoritmo *Local Interpretable Model-agnostic Explanations* (LIME) para interpretar os resultados dos modelos de classificação. Os melhores resultados para classificação ficaram com as configurações que utilizam a EfficientNet como extrator de característica e a regressão logística como classificador. Para segmentação semântica e localização de objetos obteve-se resultados com a U-net e a Yolov5, respectivamente. Os resultados encontrados demonstram a viabilidade de se utilizar os modelos de aprendizagem de máquina para automatizar o processo de separação e seleção das imagens provenientes das redes sociais, reduzindo assim o tempo empregado na análise das imagens. Deseja-se também que o trabalho auxilie outros projetos envolvendo recifes de corais da costa brasileira e demais regiões.

Palavras-chave: Redes Neurais Convolucionais, Aprendizagem de Máquinas, Visão Computacional, Ecologia Marinha

ABSTRACT

Corals are colonial animals within the Phylum Cnidaria that form coral reefs which play a major role in the marine ecosystem, providing habitat for fishes, mollusks, crustaceans, sponges, algae and other organisms. However, corals are losing coloration through the disruption of a symbiotic relationship with microalgae driven by rising temperatures and climate change, often leading to coral mortality. Due to its importance for the marine environment, monitoring the coral reefs is fundamental and nowadays is made by data collected from underwater photographs. In this context the Marine Ecology Laboratory (LECOM) from the Federal University of Rio Grande do Norte (UFRN) developed the project "#DeOlhoNosCorais" which encourages people to post photos of coral reefs on their social media. Thus, the laboratory acquires information of the health of the corals in the Brazilian coast. However, the analysis of these images is currently carried out manually, and it requires a specialist. In this dissertation, we worked in partnership with LECOM to carry out the training of machine learning models to perform the classification of the posts on social networks with the hashtag "#DeOlhoNosCorais", and then reducing the time used in the analysis of the images. A differential of the database used in the work in relation to those present in the literature is the presence of the segmentation map that classifies each individual pixel of the image. So, we evaluated the machine learning models against 3 scenarios: classification, semantic segmentation and object localization. We also tested the transfer learning using the Pacific Labeled Corals (PLC) dataset and the Local Interpretable Model-agnostic Explanations (LIME) algorithm to explain the results of the classification models. The best results for the classification task were using the EfficientNet for the feature extraction and Logistic Regression for the classification. For the semantic segmentation and object localization, we used the U-net and Yolov5 models, respectively. The result founded demonstrate the feasibility of using machine learning models to automate the process of separation and selection of the images from the social networks, thus reducing the time spent on the image analysis. It is hoped that the work will assist other projects involving coral reefs on the Brazilian coast and other regions.

Keywords: Convolutional Neural Network, Machine Learning, Computer Vision, Marine Ecology

LISTA DE FIGURAS

LISTA DE TABELAS

Lista de Acrônimos

AdaGrad	*Adaptive Gradients*
Adam	*Adaptive Moment Estimation*
APα	*Average Precison* α
AP50	*Average Precison 50*
AP95	*Average Precison 95*
AUV	*Autonomous Underwater Vehicle*
CCA	*Crustose Coralline Algae*
CE	*Cross Entropy*
CLBP	*Completed Local Binary Pattern*
CNN	*Convolutional Neural Network*
CPU	*Central Processing Unit*
DBN	*Deep Belief Network*
DINF	Departamento de Informática
EILAT	*Eilat Dataset*
EILAT 2	*Eilat 2 Dataset*
F1	*F1 score*
FCN	*Fully Convolutional Network*
FN	*False Negative*
FP	*False Positive*
GAN	*Generative Adversarial Network*
GLCM	*Grey Level Co-occurence Matrix*
GPU	*Graphics Processing Unit*
GT	*Ground Truth*
ILSVRC-2012	*ImageNet Large Scale Visual Recognition Challenge 2012*
ILSVRC-2014	*ImageNet Large Scale Visual Recognition Challenge 2014*
ILSVRC-2015	*ImageNet Large Scale Visual Recognition Challenge 2015*
IoU	*Intersection-Over-Union*
KNN	*K-Nearest Neighbors*
L	*Loss*
Lasso	*Least Angle Regression*
LECOM	Laboratório de Ecologia Marinha
LIME	*Local Interpretable Model-agnostic Explanations*
mAPα	*Mean Average Precison* α
MCC	*Matthews Correlation Coefficient*
mIoU	*Mean Intersection-Over-Union*
MLC	*Moorea Labeled Corals*
MLC-LTER	*Moorea Coral Reef-Long Term Ecological Research*
MLP	*Multilayer Perceptron*
MSRA	*Microsoft Research Asia*
PCA-SVM	*Principal Component Analysis-Support Vector Machine*
PDWMD	*Probability Density Weighted Mean Distance*
PLC	*Pacific Labeled Corals*
PSPNet	*Pyramid Scene Parsing Network*

ReLU	*Rectified Linear Unit*
RGB	*Red Green Blue*
RMSProp	*Root Mean Squared Propagation*
RNN	*Recurrent Neural Network*
RSMAS	*Rosenstiel School of Marine and Atmospheric Sciences*
SEEDS	*Superpixels Extracted via Energy-Driven Sampling*
SGD	*Stochastic Gradient Descent*
SHAP	*Shapley Additive exPlanations*
SLIC	*Simple Linear Iterative Clustering*
SVM	*Support Vector Machine*
TN	*True Negative*
TP	*True Positive*
UFPR	Universidade Federal do Paraná
UFRN	Universidade Federal do Rio Grande do Norte
XGBoost	*eXtreme Gradient Boosting*

SUMÁRIO

1 INTRODUÇÃO

Os corais são animais coloniais do Filo Cnidaria que formam os recifes de corais. Os recifes de corais servem de habitat para diversas espécies como algas, fungos, bactérias e peixes. De acordo com Chen et al. (2015), estima-se que cerca de 25% das espécies marinhas utilizam os recifes como habitat. Os corais também são importantes economicamente para regiões costeiras, movimentando cerca de 30 bilhões de dólares anualmente através da pesca e do turismo providenciado pelo ecossistema marinho. Os recifes fornecem comida e recursos para aproximadamente 500 milhões de pessoas mundialmente.

Em decorrência das mudanças climáticas, a área de ocupação dos recifes está diminuindo a uma taxa de 1-2% ao ano. Estima-se que entre os anos de 1997 e 1998 o evento de branqueamento dos corais devido ao El Niño causou um impacto econômico de 91 milhões de dólares no Caribe (Chen et al., 2015; Hoegh-Guldberg, 2011).

De acordo com Hoegh-Guldberg (1999) o aquecimento dos oceanos gera uma rápida evasão das microalgas que vivem em associação com corais, esse evento é denominado branqueamento do coral. A coloração branca (Figura 1.1) aparece, pois com a redução da população de microalgas é possível ver o esqueleto do coral através de seu tecido gelatinoso e transparente. Quando a temperatura dos oceanos aumenta os corais podem conseguir se adaptar às novas condições, no entanto em alguns casos esse evento pode levar a morte dos recifes de corais.

Figura 1.1: Exemplo do evento branqueamento para um coral da espécie *Siderastrea stellata*

O aumento da concentração de CO_2 na atmosfera também influência no ecossistema marinho. O CO_2 antropogênico[1] presente na atmosfera é depositado nos oceanos causando sua acidificação. Essa acidificação dos oceanos gera a calcificação dos recifes de corais. Outra consequência é que em alguns gêneros (*Acropora*, *Seriatopora*, *Montipora* e *Stylophora*) essa mudança química das águas oceânicas pode ocasionar a diminuição da tolerância das espécies ao aumento da temperatura das águas (Hoey et al., 2016).

Hughes et al. (2018) analisaram os dados de eventos de branqueamentos de corais em 4 regiões: Australásia[2], oceano Índico, oceano Pacífico e Atlântico ocidental. Conforme a Figura 1.2 o Atlântico ocidental começou a experienciar eventos de branqueamento primeiro que as outras regiões e nesse período a região sofreu 7 eventos que atingiram pelo menos 50% da região, em comparação a Australásia e o oceano Índico que sofreram 3 vezes e no Pacífico somente 2 vezes no mesmo período. Através dos anos o risco de branqueamento dos corais nas regiões da

[1] Produzido a partir de atividade humana

[2] Região que inclui a Austrália, Nova Zelândia, Nova Guiné e ilhas menores da parte oriental da Indonésia

Australásia e no oceano Índico vem aumentando consideravelmente, no Pacífico esse aumento é moderado e vem diminuindo no Atlântico ocidental.

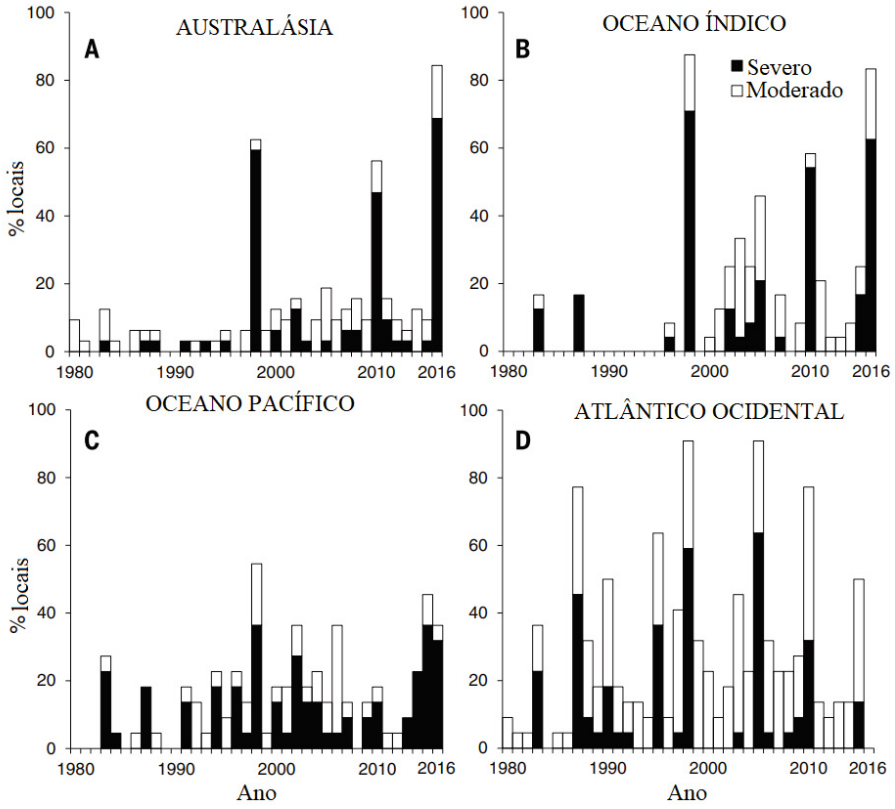


Figura 1.2: Comparação do período e intensidade do evento de branqueamento dos corais nas regiões da Australásia, oceano Índico, oceano Pacífico e Atlântico ocidental dos anos de 1980 até 2016. Fonte: Adaptado de Hughes et al. (2018)

Devido à importância dos recifes de corais para o ecossistema marinho, seu monitoramento é essencial para evitar impactos ambientais e econômicos. Atualmente, o monitoramento normalmente acontece com fotos registradas por mergulhadores. Em regiões onde o mergulho é restrito utiliza-se *Autonomous Underwater Vehicles* (AUVs). Essas fotos são armazenadas e analisadas por especialistas em ecologia marinha a fim de identificar possíveis inconformidades.

O trabalho atual conta com a parceria da Universidade Federal do Rio Grande do Norte (UFRN) que possui o projeto "#DeOlhoNosCorais" que visa aumentar o monitoramento da saúde dos recifes de corais incentivando usuários de redes sociais a postarem fotos de recifes de corais fotografados com seus equipamentos pessoais e utilizando a *hashtag* do projeto. Com essa base de dados gerada por usuários sem treinamento científico prévio, os pesquisadores analisam a saúde dos corais na costa brasileira.

A análise das imagens dos recifes de corais é normalmente feita manualmente por um especialista em ecologia marinha. A fim de automatizar o processo de análise pode-se também utilizar algoritmos de aprendizagem de máquina. Os modelos conseguem auxiliar a equipe de pesquisa nas tarefas de análise preliminar, identificação de espécies e reconhecimento de padrão.

Esses modelos podem atualmente classificar e segmentar a imagem criando seu mapa de segmentação que é a classificação de todos os píxeis presentes na imagem ou localizando objetos que adiciona *bounding boxes* no entorno dos objetos presentes da imagem. A partir de um modelo treinado os pesquisadores poderiam rapidamente selecionar apenas as espécies de interesse.

1.1 MOTIVAÇÃO

Na literatura existem trabalhos que realizaram a classificação dos recifes de corais utilizando aprendizagem de máquina (Shihavuddin et al. (2013), Gómez-Ríos et al. (2019), Lumini et al. (2019)). Com o desenvolvimento de novos modelos de aprendizagem de máquina utilizando redes neurais convolucionais, também se desenvolveu trabalhos que realizam a segmentação semântica dos recifes de corais utilizando esta técnica (Alonso et al. (2017), King et al. (2018)). Entretanto, nos trabalhos de Alonso et al. (2017) e King et al. (2018) não foram utilizados mapas de segmentação criados por especialistas, foram sintetizados mapas de segmentação a partir das anotações presentes na imagem.

O presente trabalho conta com a primeira versão da base de dados disponibilizada pelos nossos parceiros que contém imagens extraídas de rede sociais com a hashtag "#DeOlhoNosCorais" e imagens cedidas pelo Laboratório de Ecologia Marinha (LECOM) da UFRN. A atual base de dados contém 1411 imagens e 21 classes com seus respectivos mapas de segmentação. A presença dos mapas de segmentação é um diferencial em relação às bases de dados disponíveis na literatura que normalmente possuem somente anotações em 1 píxel da imagem. No mapa de segmentação, como pode ser visto na Figura 1.3, existe a localização completa do coral na imagem.

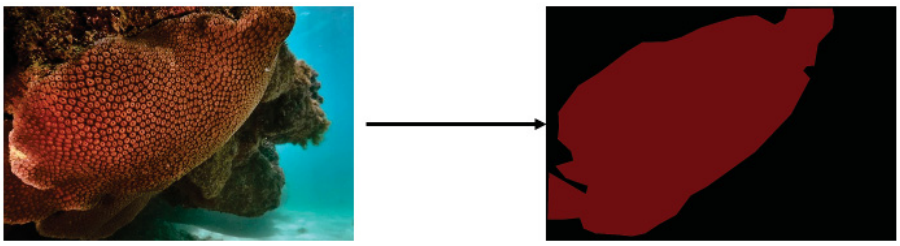


Figura 1.3: Exemplo do mapa de segmentação para uma imagem que contém um coral da espécie *Montastraea cavernosa*

Na atual versão da base não existe a rotulação em relação à condição do coral, por exemplo: vivo, morto, branqueado. Essa rotulação pode ser alvo de trabalhos futuros que se propõem em treinar modelos de aprendizagem de máquina para classificar a condição dos corais presentes nas imagens.

Um dos desafios presentes na base de dados é a existência de ruídos (textos, marca d'água), padrões de cores diferentes (saturação, balanço de branco, brilho) e resoluções diferentes nas imagens, como pode ser visto na Figura 1.4, já que essas são fotografadas por usuários distintos, com equipamentos variados e podem sofrer algum tipo de tratamento antes de serem postadas. Os trabalhos na literatura em geral utilizam base de dados retiradas de fontes únicas e, consequentemente, os dados são mais padronizados. Outro desafio é a escassez de trabalhos que utilizam aprendizagem de máquina voltados para recifes de corais presentes na costa brasileira.

1.2 PERGUNTAS DE PESQUISA

Pensando nesse contexto de recifes de corais e nas dificuldades de análise dos dados, o atual trabalho possui as seguintes perguntas de pesquisa:

1. É viável utilizar aprendizagem de máquinas para automatizar o processo de análise das imagens com a *hashtag* "#DeOlhoNosCorais"?

Figura 1.4: Três imagens presentes na base de dados cedida pelo LECOM da UFRN

2. Utilizar *transfer learning* de outras bases de dados de corais, melhora o desempenho do classificador?

3. Quais são os locais da imagem mais relevantes para os modelos realizarem a classificação?

1.3 OBJETIVOS

O objetivo principal do trabalho é avaliar a viabilidade de se utilizar modelos de aprendizagem de máquina para automatizar o processo de análise das imagens com a *hashtag* "#DeOlhoNosCorais". Portanto, os modelos de aprendizagem de máquina foram avaliados frente a 3 cenários:

1. Imagem inteira (classificação utilizando as imagens originais)

2. Sub-imagens da imagem inteira (classificação utilizando imagens recortadas da imagem inteira)

3. Segmentação semântica e localização de objetos

O primeiro cenário, avaliou o desempenho do modelo utilizando as imagens inteiras, sendo assim as imagens permaneceram como foram retiradas das redes sociais. As predições foram interpretadas usando um algoritmo que avalia quais locais das imagens são mais relevantes para os modelos. Os desafios dessa etapa foram o baixo número de imagens disponíveis e os ruídos presentes nas imagens como apresentado na Figura 1.4.

No segundo cenário foram retiradas sub-imagens dos corais para reduzir o ruído das imagens. Consequentemente, essa extração aumentou o número de imagens da base de dados. Por fim, avaliou-se o impacto dos ruídos e do número de imagens no desempenho dos classificadores.

Para minimizar o problema referente ao número de imagens presentes na base de dados, testou-se o impacto de se utilizar *transfer learning* com outras bases de dados no desempenho dos classificadores. Avaliou-se duas metodologias, a primeira utilizou-se somente a base ImageNet (Deng et al., 2009). Em contrapartida, para a segunda metodologia empregou-se uma combinação da ImageNet com a base *Pacific Labeled Corals* (PLC) como intermediária.

No terceiro cenário realizou-se testes iniciais com modelos de segmentação semântica e localização de objetos com rótulos binários para ambas as tarefas. Deseja-se com esses testes demonstrar a possibilidade de se utilizar modelos que conseguem encontrar os corais nas imagens. De forma geral, deseja-se que o trabalho auxilie futuros projetos, pesquisas e análises envolvendo recifes de corais na costa brasileira e em outras regiões.

1.4 CONTRIBUIÇÕES

Levando em considerações os resultados encontrados e a metologia desenvolvida, as principais contribuições alcançadas pelo trabalho são as seguintes:

- Criação da base de dados[3] com imagens de corais brasileiros em parceria com os pesquisadores do LECOM da UFRN.
- Desenvolvimento de uma metodologia para a utilização da base de dados para treinamentos de modelos de aprendizagem de máquina para as tarefas de classificação, localização de objetos, segmentação semântica e segmentação instanciada. Facilitando assim a reprodutibilidade dos resultados encontrados e servindo de suporte para trabalhos futuros.
- Apresentação de resultados para as principais tarefas de visão computacional: classificação, segmentação semântica e localização de objetos. Esses resultados são relevantes para o contexto de recifes de corais, devido ao baixo número de trabalhos que correlacionam aprendizagem de máquina e recifes de corais.
- Primeiros resultados utilizando *transfer learning* entre bases de recifes de corais
- Primeiros resultados para a segmentação semântica de corais utilizando mapas de segmentação sem aproximações.
- Primeiros resultados utilizando aprendizagem de máquina para analisar imagens de corais brasileiros.

[3] https://doi.org/10.5281/zenodo.7338208

2 FUNDAMENTAÇÃO TEÓRICA

Um algoritmo de aprendizagem de máquina é definido como um algoritmo que consegue aprender a partir de dados. O termo "aprender" pode ser interpretado de várias formas. De acordo com Mitchell (1997) é definido como "Uma aplicação consegue aprender se a partir de uma experiência E com algumas classes da tarefa T e métrica de performance P o seu desempenho na tarefa T, medida por P, melhora com a experiência E".

A experiência E pode ser dividida em duas categorias: o aprendizado supervisionado e o não supervisionado. No aprendizado supervisionado cada dado possui um rótulo/classe. Por exemplo, em uma base de dados para reconhecimento de cachorros e gatos (Elson et al., 2007) cada imagem possui sua respectiva classe como pode ser visto na Figura 2.1.

Figura 2.1: Exemplo de uma base de dados para aprendizado supervisionado que contém imagens rotuladas de cachorros e gatos. Fonte: Adaptado de TensorFlow (2021)

No aprendizado não supervisionado não existem as classes dos dados, então o aprendizado é efetuado através das características presentes na base de dados. O algoritmo realiza a separação dos exemplos em grupos semelhantes (*clustering*). Um exemplo de aprendizado não supervisionado é sistemas de recomendação que separa os usuários de uma plataforma em grupos de interesses semelhantes.

Atualmente, os modelos de aprendizagem de máquina desempenham várias tarefas T, como reconhecimento de faces (Wang e Deng, 2018), detecção de objetos (Liu et al., 2018), reconhecimento de fala (Malik et al., 2020), processamento de linguagem natural (Torfi et al., 2020). Com o avanço na capacidade de processamento das *Graphics Processing Units* (GPUs) e o aumento do número de dados disponíveis, os modelos de aprendizagem de máquinas que utilizam *deep learning* (Seção 2.2) ganharam popularidade na última década (Mahapatra, 2018).

2.1 CAPACIDADE, OVERFITTING, UNDERFITTING

No treinamento de um modelo de aprendizagem de máquina espera-se que ele consiga aprender a partir da base de treinamento e obtenha resultados também em uma base de dados de teste que não é utilizada para ajustar os parâmetros do modelo. A habilidade do modelo de performar bem nas duas bases de dados (treino e teste) é denominada generalização. Modelos que possuem uma boa generalização conseguem transpor o "conhecimento" adquirido na fase de treinamento para outros ambientes (fase de teste). Logo, deseja-se que a diferença entre o erro na base de treino e de teste seja minimizada.

A capacidade de um modelo é dita como a habilidade de se adequar a várias funções. Modelos com baixa capacidade (Figura 2.2(a)) são propensos a *underfitting* que é a incapacidade de obter bom desempenho na base de treinamento. Modelos com alta capacidade (Figura 2.2(c)) se adéquam muito a base de treinamento gerando baixa generalização (*overfitting*). A Figura 2.2(b) apresenta um modelo com capacidade ideal.

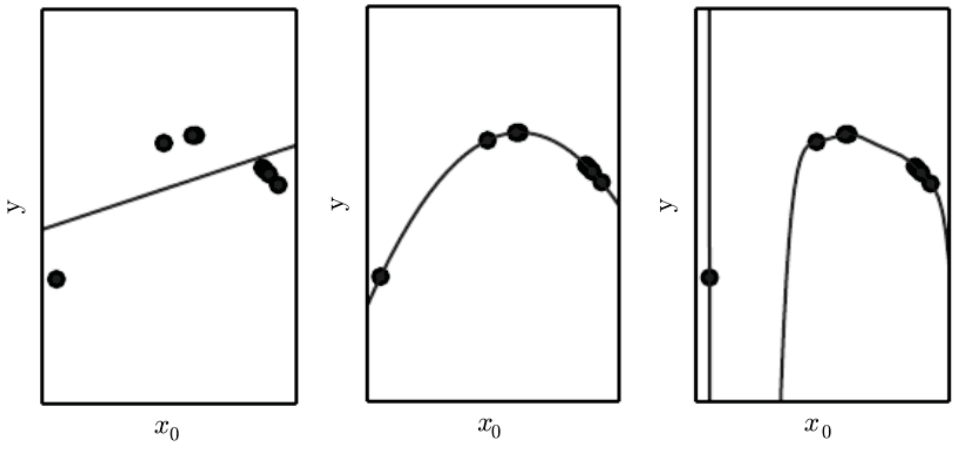


Figura 2.2: Comparação da generalização de três modelos utilizando pontos sintéticos. (a) Modelo em *underfitting*, (b) modelo com generalização ideal, (c) modelo em *overfitting*. Fonte: (Goodfellow et al., 2016)

2.2 DEEP LEARNING

Modelos que utilizam *deep learning* (aprendizado profundo) são uma das categorias de algoritmos de aprendizagem de máquinas. Existem outros modelos como *Support Vector Machine* (SVM) (Cortes e Vapnik, 1995), Random Forests (Tin Kam Ho, 1995), *eXtreme Gradient Boosting* (XGBoost) (Chen e Guestrin, 2016) que utilizam outras metodologias para seu aprendizado.

O aprendizado profundo surgiu após o desenvolvimento do algoritmo de *back-propagation* (Rumelhart et al., 1986). O algoritmo possibilitou o treinamento do *perceptron* (Rosenblatt, 1958) utilizando múltiplas camadas. O modelo com múltiplas camadas é denominado *Multilayer Perceptron* (MLP). A Figura 2.3 ilustra a diferença do *perceptron* (Figura 2.3(a)) e do MLP (Figura 2.3(b)).

Os MLPs são formados por diferentes camadas para que a partir de uma entrada seja gerado uma saída. Essas cadeias de células interligadas lembram as conexões cerebrais, por isso essas estruturas são chamadas também de redes neurais.

O modelo MLP são o modelo primordial para entender os modelos de aprendizagem profunda. Com o passar dos anos outras variações de redes neurais surgiram como as *Convolu-*

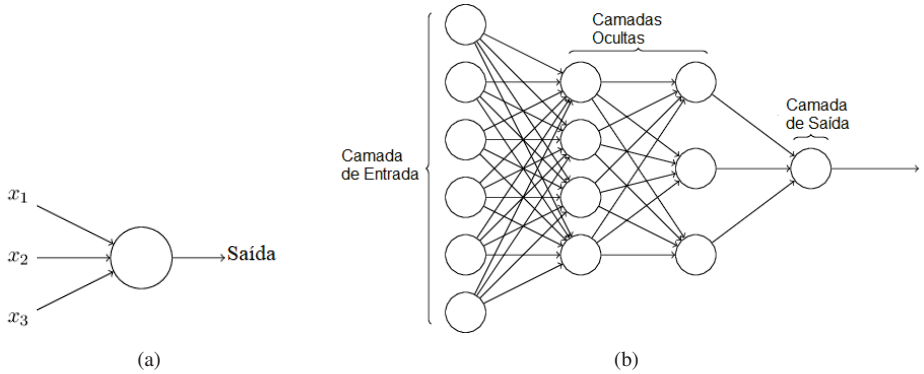


Figura 2.3: Diferença entre o (a) *Perceptron* e um (b) MLP com 4 camadas. Fonte: Adaptado de Rodriguez (2020); Academy (2022)

tional Neural Networks (CNNs), *Recurrent Neural Networks* (RNNs) e *Deep Belief Networks* (DBNs).

A profundidade de uma rede neural é definida pelo número de camadas da rede. As camadas são classificadas em camadas de entrada, ocultas e de saída. As camadas de entrada e saída são respectivamente a primeira e a última camada da rede neural e as camadas ocultas são as camadas intermediárias.

As células em um MLP são *perceptrons*. A Equação 2.1 apresenta o cálculo que cada célula realiza para gerar uma saída (h) a partir de uma entrada (x). Onde, W são os pesos (*weights*), b é o viés (*bias*) e g é a uma função de ativação (Seção 2.5). Durante a fase de treinamento os valores de W e b são atualizados conforme os dados entregues ao modelo.

$$h = g(W^\top x + b) \tag{2.1}$$

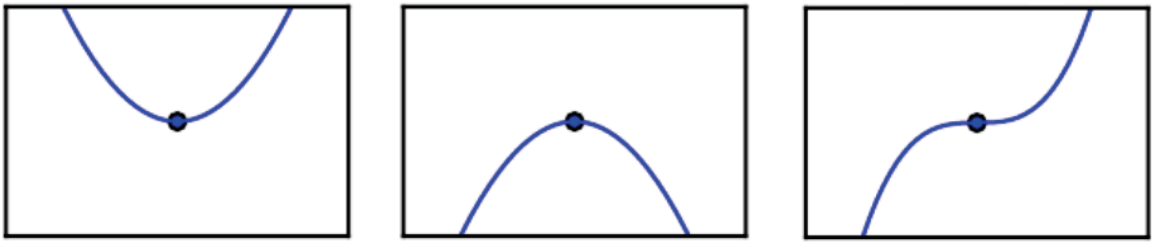
Os modelos que utilizam aprendizagem profunda para o seu treinamento possuem uma função de custo (Seção 2.4) que computa a qualidade da predição para a entrada x e um otimizador (Seção 2.6) que irá atualizar os parâmetros visando minimizar a função de custo. Devido a função de custo e as camadas de não linearidade (funções de ativação) o problema de otimização de uma rede neural não é convexo, portanto se utiliza a metodologia de descida do gradiente (Seção 2.3) para atualizar os parâmetros.

2.3 DESCIDA DO GRADIENTE

A descida do gradiente proposto por CAUCHY (1847) utiliza o gradiente de uma função para realizar a sua otimização. Seja uma função $y = f(x)$, onde x e y são números reais. Sua derivada $f'(x)$ retorna o valor de inclinação de $f(x)$ dado um ponto x. Esse valor demonstra como a função se comporta no ponto x se x sofrer uma pequena alteração ϵ, $f(x + \epsilon) = f(x) + \epsilon f'(x)$.

Se $f(x)' = 0$, o ponto x é considerado um ponto crítico ou ponto estacionário da função. O ponto crítico (Figura 2.4) pode corresponder a 3 diferentes situações. O ponto pode ser um mínimo local (Figura 2.4(a)) que o valor de $f(x)$ é inferior aos seus vizinhos. Um máximo local (Figura 2.4(b)) que o valor de $f(x)$ é superior aos dos seus vizinhos. Pode ser um ponto de sela que o ponto não é um mínimo local ou máximo local.

No caso que a função possui múltiplas variáveis utiliza-se a derivada parcial de f. A derivada parcial $\frac{\partial}{\partial x_i} f(x)$ avalia como f muda se somente a variável x_i mudar no ponto x. O gradiente da função generaliza o conceito da derivada para os casos onde a derivada é respectiva a um vetor. O gradiente de f é um vetor contendo todas as derivadas parciais, denotado $\nabla_x f(x)$.



(a) (b) (c)

Figura 2.4: Diferença nos tipos de ponto crítico - (a) mínimo (b) máximo (c) ponto de sela. Fonte: (Goodfellow et al., 2016)

O gradiente da função f aponta diretamente para onde a função aumenta, portanto o gradiente negativo aponta para onde a função reduz. Para encontrar o ponto onde todas as derivadas parciais são iguais a zero, utiliza-se o método de descida do gradiente que move f na direção do gradiente negativo em pequenos passos (Equação 2.2). Onde, x_t é o valor de x no instante de tempo t, x_{t+1} é o valor de x no próximo passo, ϵ é a taxa de aprendizado (*learning rate*), e $\nabla_x f(x_t)$ é o gradiente da função.

$$x_{t+1} = x_t - \epsilon \nabla_x f(x_t) \tag{2.2}$$

O algoritmo de *back-propagation* foi desenvolvido para computar o gradiente da rede neural (Rumelhart et al., 1986). O algoritmo utiliza a regra da cadeia (Leibniz (1676); L'Hôpital (1696)) para medir a influência das variáveis na rede neural sobre a função de custo. Seja x um número real, com f e g ambas funções reais. Supondo que $y = g(x)$ e $z = f(g(x)) = f(y)$. A regra da cadeia afirma que:

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx} \tag{2.3}$$

Onde, $\frac{dz}{dx}$ é a derivada de z em relação a x, $\frac{dz}{dy}$ é a derivada de z em relação a y e $\frac{dy}{dx}$ é a derivada de y em relação a x. Dessa forma, a regra da cadeia possibilita a avaliação do gradiente das células presentes na rede neural da camada saída até a camada de entrada passando pelas camadas ocultas.

2.4 FUNÇÃO DE CUSTO

A função de custo avalia a predição do modelo para cada classe do problema. O seu resultado é o principal fator para a atualização dos pesos e vieses da rede neural. A função é calculada após os resultados extraídos da camada de saída.

As principais funções de custo para classificação são a *Softmax cross entropy* e a *Sigmoid cross entropy*. Para tarefa de classificação utiliza-se a probabilidade gerada pelo modelo para cada classe do problema.

Utiliza-se a *Sigmoid cross entropy* como função de custo para saídas binárias (2 classes). Para o seu cálculo utiliza-se a função de ativação *Sigmoid* (Equação 2.4) sobre os resultados da camada de saída (x) para a classe $y = 1$ normalizando os resultados no intervalo entre $[0, 1]$. Como a soma das probabilidades deve ser 1, a segunda classe $y = 0$ tem probabilidade igual a $P(y = 0 \mid x) = 1 - P(y = 1 \mid x)$.

$$\hat{y} = P(y = 1 \mid x) = \frac{1}{1 + e^{-x}} \tag{2.4}$$

Aplicando a *Cross Entropy* (CE) (Equação 2.5) entre as distribuições de probabilidade p e q, onde p é a distribuição verdadeira e q é a distribuição predita para todas as classes i presentes na tarefa.

$$CE = -\sum_{i} p_i \log(q_i) \tag{2.5}$$

Utilizando as Equações 2.4 e 2.5 obtém-se a *Sigmoid cross entropy* (Equação 2.6) no caso $p \in (y, 1 - y)$ e $q \in (\hat{y}, 1 - \hat{y})$, onde y é a classe correta (0 ou 1) da entrada x. O resultado da função de custo é denominado perda, mas também é conhecida popularmente pelo nome em inglês *Loss* (L).

$$L = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}) \tag{2.6}$$

Utiliza-se a *Softmax cross entropy* para saídas categóricas (3 ou mais classes). A função de ativação da camada de saída é a *Softmax* (Equação 2.7) também conhecida como normalização exponencial realiza normalização ($\hat{y}_i$) da predição do modelo x_i para uma classe y_i em função da predição do modelo para todas as outras classes ($\sum_{j=1}^{K} x_j$). Onde, K é número de classes, x é a predição do modelo para cada classe $x = (x_1, ..., x_K) \in \mathbb{R}^K$.

$$\hat{y}_i = P(y = y_i \mid x) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{2.7}$$

Para calcular a perda utiliza-se a CE (Equação 2.5) e a *Softmax* (Equação 2.7). Onde, a distribuição p no caso de classificação categórica assume valor 1 quando i é a classe correta e 0 para as classes incorretas. Portanto, a *Softmax cross entropy* é definida como:

$$L = -\log\left(\frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}\right) \tag{2.8}$$

Utilizando logaritmo natural para desfazer a exponencial do numerador a *Softmax cross entropy* fica:

$$L = -x_i + \ln\left(\sum_{j} e^{x_j}\right) \tag{2.9}$$

A *Softmax* possui uma propriedade interessante para o treinamento das redes neurais, pois se considera a probabilidade de todas as classes para o seu cálculo. Para se atingir resultados de ($\hat{y} \cong 1$) o modelo precisa obter um resultado para classe correta i muito superior às demais classes.

2.5 FUNÇÕES DE ATIVAÇÃO

O objetivo das camadas ocultas é de extrair características da entrada. As funções de ativação procuram realizar transformações matemáticas ao resultado da célula para acrescer não

linearidade ao modelo e quantificar a taxa de ativação da célula. As funções de ativação mais populares são a *Sigmoid*, a tanh e a *Rectified Linear Unit* (ReLU) (Figura 2.5).

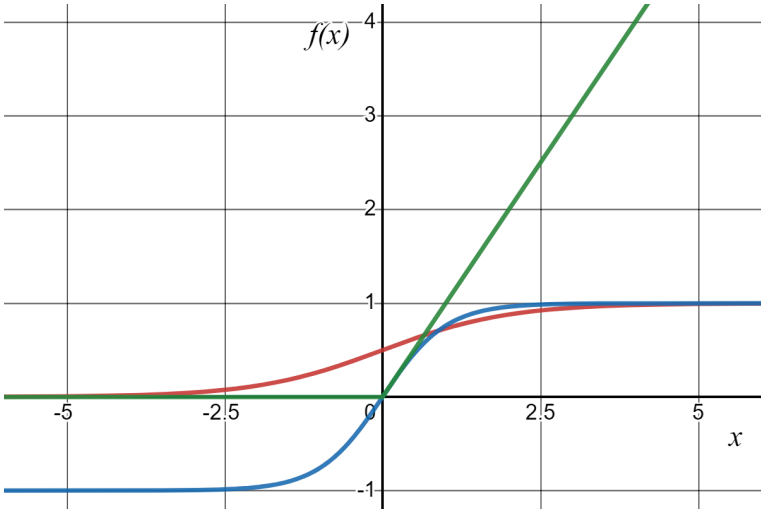


Figura 2.5: Comparação das funções de ativação para valores de x entre -5,5 e 5,5. *Sigmoid* (Vermelho), tanh (Azul), ReLU (Verde)

A função *Sigmoid* (Equação 2.4) reduz o resultado das células a valores entre 0 e 1 ela pode ser interpretada como a escala de ativação da célula. Atualmente essa função de ativação vem caindo em desuso, devido aos casos de saturação do neurônio, quando não há ativação (0) ou quando há ativação máxima (1). Nesses casos ocorre o desparecimento do gradiente impossibilitando o treinamento. Outro ponto é que a *Sigmoid* não possui o ponto 0 como centro.

A função de ativação tanh ($f(x) = tanh(x)$) ativa os neurônios no intervalo entre $[-1, 1]$. Os pontos positivos da tanh em relação a *Sigmoid* é que ela é centrada no ponto 0, entretanto ela também possui dois pontos de saturação no -1 e no 1.

A ReLU ($f(x) = max(0, x)$) é a função de ativação mais popular atualmente, principalmente depois do desenvolvimento da AlexNet (Krizhevsky et al., 2012). Os pontos fortes da ReLU são sua baixa demanda computacional e a existência de uma boa propagação do gradiente devido a não haver limitação quando $x > 0$. Entretanto, algumas células podem ser atualizadas de uma determinada forma que elas nunca serão ativadas durante a etapa de treinamento. Em alguns casos pode ocorrer a explosão do gradiente onde os valores de ativação são muito altos.

2.6 OTIMIZADORES

O *Stochastic Gradient Descent* (SGD) é uma variação da descida do gradiente (Equação 2.2). No SGD os exemplos de treinamento são separados em *minibatches* $\mathbb{B} = (x^{(1)}, ..., x^{(N)})$ e o cálculo do gradiente g é a média do gradiente de todos os exemplos dentro de uma *minibatch*. Sendo que $\nabla_\theta L$ é o gradiente da função de custo em relação ao parâmetro do modelo θ, x são as entradas de um exemplo i contido na *minibatch*, e $f(x^i, \theta)$ é a predição do modelo e y é a classe real do exemplo i.

$$g = \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta L(f(x^i, \theta), y^i) \tag{2.10}$$

O tamanho do *batch* N é normalmente um número pequeno, variando de 1 até algumas centenas. A utilização do SGD em relação à descida do gradiente reduz o custo computacional para

o treinamento do modelo, principalmente em base de dados grandes, pois o custo computacional fica em função do tamanho do *batch*. Quando o modelo passa por todas as *minibatches* ele completa uma época, pois se passou por todos os dados de treinamento.

Modificando a equação 2.2 utilizando o gradiente resultante da Equação 2.10. A atualização dos parâmetros (θ) do modelo para cada novo passo $t + 1$ segue a forma:

$$\theta_{t+1} = \theta_t - \epsilon g \tag{2.11}$$

Para alguns hardwares é recomendado utilizar determinados tamanhos do *batch*. Em GPUs é comum se utilizar tamanho do *batch* que são base de 2, variando de 32 a 256, utilizando 16 em alguns modelos com um número de parâmetros maiores.

De acordo com Wilson e Martinez (2003), utilizar pequenos valores para o tamanho do *batch* tem um efeito regularizador. Entretanto, devido ao pequeno tamanho do *batch*, é necessário um valor também pequeno da taxa de aprendizado para manter a estabilidade do treinamento devido à alta variância na estimativa do gradiente. Essas alterações também impactam o tempo de treinamento, exigindo um tempo maior para o treinamento do modelo.

O otimizador SGD é bem popular, entretanto ele pode exigir um alto tempo de treinamento. O método do *momentum* foi desenvolvido por Polyak (1964) para acelerar o treinamento. O algoritmo acrescenta uma nova variável v que se comporta como a velocidade. No algoritmo de treinamento com *momentum*, o novo gradiente (v) é calculado com valores de gradientes passados (v_{t-1}) para manter o seu "*momentum*" através do espaço de parâmetros. O modelo contém também um parâmetro $\alpha \in [0, 1)$ que determina o fator de contribuição dos gradientes passados.

Acrescentando os parâmetros v_{t-1} e α na Equação 2.10 obtém-se a nova regra de atualização do gradiente (v) no instante de tempo t (Equação 2.12). A atualização dos parâmetros apresentada na Equação 2.13 segue a mesma regra da Equação 2.11.

$$v_t = \alpha v_{t-1} - \epsilon \, (\frac{1}{N} \sum_{i=1}^{N} \nabla_\theta L(f(x^i, \theta), y^i)) \tag{2.12}$$

$$\theta_{t+1} = \theta_t + v_t \tag{2.13}$$

A Figura 2.6 ilustra a comparação entre o SGD e o SGD + *momentum*, as setas pretas na Figura 2.6(b) representam o passo que o algoritmo SGD tomaria naquele instante de tempo. Observa-se que o otimizador SGD + *momentum* reduz o número de passos de treinamento.

Em muitos casos no otimizador SGD é necessário a definir também um decaimento da taxa de aprendizado (ϵ) durante o treinamento. Um método comum é o decaimento exponencial (Equação 2.14). Onde, λ é o parâmetro que ajusta a magnitude do decaimento, t é o tempo e ϵ_0 é a taxa de aprendizado inicial.

$$\epsilon(t) = \epsilon_0 - e^{-\lambda t} \tag{2.14}$$

Na última década, desenvolveram-se alguns otimizadores denominados otimizadores adaptáveis que alteram a taxa de aprendizado conforme o curso do treinamento. Atualmente, os mais populares são o *Root Mean Squared Propagation* (RMSProp) (Hinton, 2012) e o *Adaptive Moment Estimation* (Adam) (Kingma e Ba, 2017).

O RMSProp modifica o algoritmo *Adaptive Gradients* (AdaGrad) (Duchi et al., 2011) para aperfeiçoar a convergência em problemas não convexos, alterando a acumulação de gradiente em uma média móvel exponencialmente ponderada. O algoritmo Adam é uma modificação do

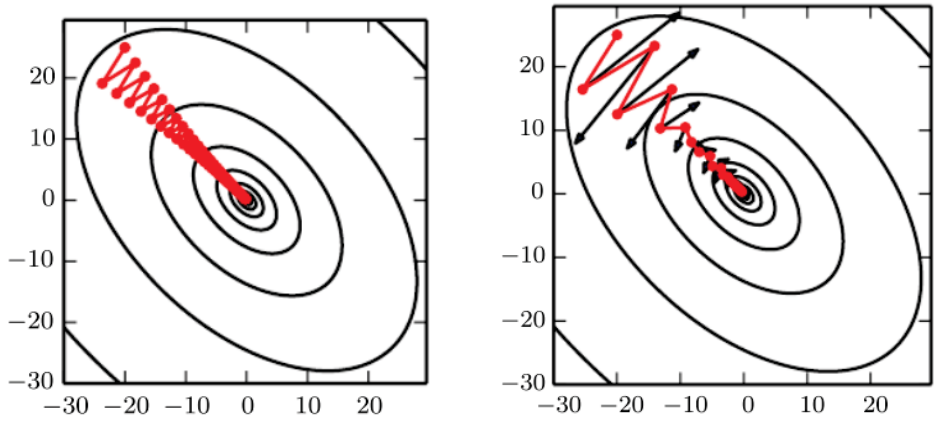


Figura 2.6: Comparação da convergência entre os otimizadores (a) SGD e (b) SGD + *momentum*. Os pontos vermelhos indicam os passos dos otimizadores e as setas pretas no SGD + *momentum* indica qual seria o próximo passo se não tivesse o parâmetro de *momentum*. Fonte: (Goodfellow et al., 2016)

RMSProp. O Adam inclui um fator de correção para estimar ambos *momentum* de primeira ordem e de segunda ordem. O RMSProp também possui um estimador de um *momentum* de segunda ordem, porém não possui o fator de correção (Goodfellow et al., 2016).

2.7 REGULARIZADORES

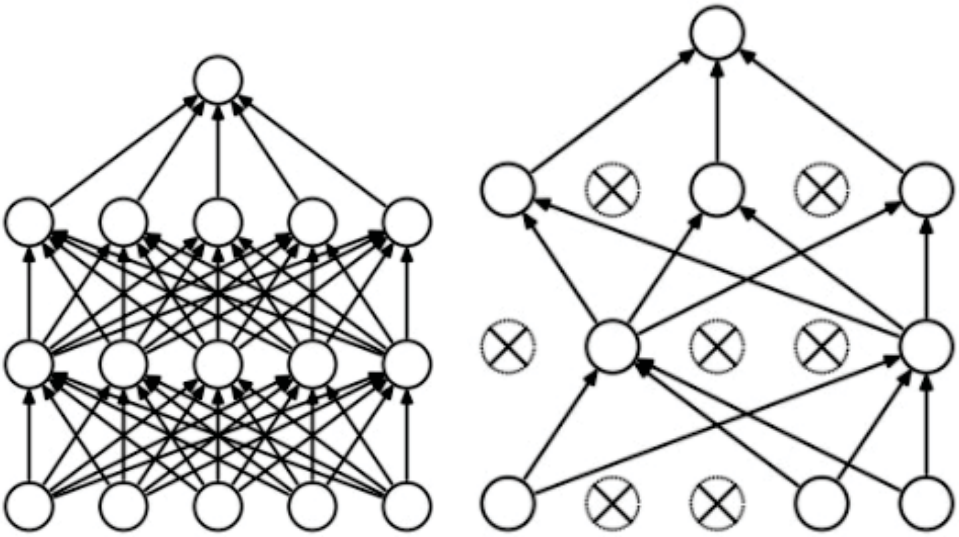
Na seção 2.1 discutiu-se os conceitos de capacidade, *overfitting* e *underfitting* e a importância de se construir um algoritmo que obtenha resultados nas bases de treinamento e teste. Os regularizadores são estratégias que auxiliam o treinamento dos modelos de *deep learing* visando reduzir o erro durante o teste, entretanto em troca aumentam o erro durante o treino, entretanto em casos onde existe a presença de *overfitting* essa troca é benéfica.

A regularização conhecida como L^2 ou *weight decay* introduz um fator ao cálculo da função de custo $\frac{1}{2}\alpha w^2$. Onde, α controla a força de regularização. O objetivo dessa regularização L^2 é penalizar pesos (w) que possuem picos de magnitudes, conduzindo os pesos a valores mais próximos da origem.

A regularização utilizando *Dropout* (Srivastava et al., 2014) adiciona um parâmetro p que controla a probabilidade de uma célula ficar ativada ou não durante a fase de treinamento (Figura 2.7). Dessa forma, remove-se aleatoriamente um ou mais células durante cada época de treinamento. Durante a fase de teste todas as células são ativadas, pois não é desejável a introdução de aleatoriedade durante a fase de teste.

2.8 *CONVOLUTIONAL NEURAL NETWORK* (CNN)

Redes neurais convolucionais ou CNNs são especializadas em processar dados matriciais que possuem relações com seus "vizinhos", por exemplo, imagens ou séries temporais. O nome "redes neurais convolucionais" indica que a rede aplica a operação matemática chamada de convolução. Convolução é um tipo de operação linear. Redes neurais convolucionais são simplesmente redes neurais que usam convolução no lugar da multiplicação de matriz padrão (Equação 2.1) em pelo menos uma das suas camadas (Goodfellow et al., 2016).



(a) (b)

Figura 2.7: Demonstração do funcionamento do *Dropout*. As células com "x" representam células desativadas. (a) MLP com 4 camadas, (b) um exemplo de aplicação de *Dropout* no MLP apresentado na Figura (a). Fonte: (Srivastava et al., 2014)

A convolução é uma operação de somatório do produto entre duas funções, ao longo da região em que elas se sobrepõem, em razão do deslocamento existente entre elas. É uma operação popularmente utilizada quando para se reduzir o grau de ruído de uma medida.

Utiliza-se uma função de ponderamento $w(a)$, onde a é a idade de medida, aplicado as medidas $x(t)$, onde t é o tempo contínuo. Portanto, se aplicarmos a operação em todo instante de tempo encontra-se uma função mais suave s definida como:

$$s(t) = (x * w)(t) = \int x(a)w(t-a)\,da \tag{2.15}$$

Na terminologia das CNNs, o primeiro argumento, função x, é a entrada (*input*) e o segundo argumento, função w, é o *kernel* da convolução. A saída é referida como o vetor de características (*feature map*). A Equação 2.15 também pode ser utilizada para funções que não são avaliadas em um tempo contínuo e sim em tempos discretos. Portanto, assumindo que t assume valores inteiros, a convolução discreta é definida como:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{2.16}$$

Nas aplicações de aprendizagem de máquina normalmente se trabalha com entradas e *kernels* multidimensionais e as operações acontecem em vários eixos simultaneamente. No caso de uma CNN o tempo referido na equação 2.16 é substituído por pontos relativos da entrada. Por exemplo, para uma imagem I de duas dimensões como entrada com pontos relativos i e j e um kernel K de duas dimensões (m x n), obtém-se:

$$s(i,j) = (K * I)(i,j) = \sum_{m}\sum_{n} I(i-m, j-n)K(m,n) \tag{2.17}$$

As bibliotecas de redes neurais utilizam a implementação da função denominada *cross-correlation* (Equação 2.18) parecida com a convolução, entretanto as bibliotecas denominam a operação como convolução.

$$s(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n) \tag{2.18}$$

A Figura 2.8 , é um exemplo da Equação 2.18 aplicada a uma matriz de duas dimensões. O movimento do *kernel* ao longo da entrada é definido como *stride*. Nas camadas de convolução também é adicionado um viés b ao final da convolução como no *perceptron* (Equação 2.1).

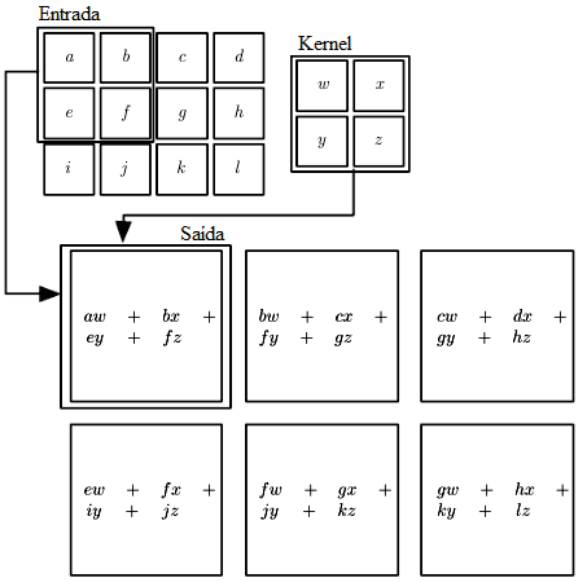


Figura 2.8: Exemplo de uma convolução com *stride* 1 utilizando uma entrada (3 x 4), um *kernel* (2 x 2), gerando uma saída (2 x 3). Fonte: Adaptado de Goodfellow et al. (2016)

De acordo com Krizhevsky et al. (2012) os *kernels* são especialistas em detecção de determinados tipos de bordas (Figura 2.9) portanto as regiões que possuem o mesmo padrão apresentam resultados de saída maiores.

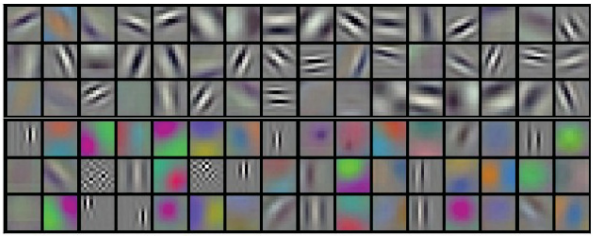


Figura 2.9: Representação visual dos *kernels* de convolução. Fonte: (Krizhevsky et al., 2012)

A especialidade em reconhecer padrões das CNNs possibilita a utilização de *transfer learing* em seu treinamento. O *transfer learning* é definido como o treinamento de um modelo para uma tarefa P_2 que foi anteriormente treinado para realizar uma tarefa P_1. Em visão computacional é comum utilizar essa metodologia com CNNs pré-treinadas na ImageNet (Oquab et al., 2014) (P_1) para retreiná-las em outra base (P_2) aproveitando os parâmetros do primeiro treinamento.

Utilizar o *transfer learning* pode viabilizar o treinamento de CNNs em bases que possuem um número pequeno de imagens. De acordo com Yosinski et al. (2014), utilizar *transfer*

learning mesmo para bases que sejam significativamente diferentes é mais benéfico que uma inicialização aleatória da CNN.

Outra operação presente nas CNNs é a de *pooling*. Essa operação resume as informações extraídas nas etapas de convolução. A função de *pooling* se comporta como a operação de convolução, porém visando reduzir a dimensão da entrada e sumarizar suas informações, deixando a rede menos invariante a pequenas alterações e aumentando a eficiência computacional. As operações comuns de se realizar na região são de max (maior valor), média, norma L^2 e média ponderada. A Figura 2.10 ilustra a operação de *max pooling*.

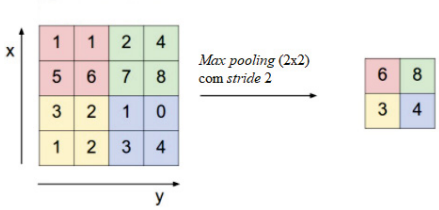


Figura 2.10: Exemplo de *max pooling* (2 x 2) com *stride* 2 para uma entrada (4 x 4) gerando uma saída (2 x 2). Fonte: Adaptado de CS231n (2020)

2.8.1 Evolução das arquiteturas de CNNs

As tarefas em visão computacional se desenvolveram ao longo dos anos a fim de que os modelos de aprendizagem de máquinas conseguissem retornar inferências mais detalhadas de imagens (Figura 2.11). As arquiteturas, a partir de um conjunto de classes $Y = \{y_1, y_2, ..., y_n\}$ e um conjunto de imagens $X = \{x_1, x_2, ..., x_n\}$, podem realizar as tarefas de classificação que retornam um ou mais classes para cada imagem (Figura 2.11(a)) ou localizar objetos com uma determinada classe colocando caixas em seu entorno (Figura 2.11(b)).

Com o intuito de conseguir apresentar resultados que auxiliam no entendimento completo da imagem, também existem modelos de aprendizagem de máquinas desenvolvidos para tarefa de segmentação que classificam cada píxel da imagem no conjunto de classes Y e com a adição da classe y_0 que é denominado plano de fundo ou *void*. Essa tarefa tem aplicações em: carros autônomos (Cordts et al., 2016), interação homem-máquina (Oberweger et al., 2015), análises médicas (Milletari et al., 2016).

A diferença entre a segmentação semântica (Figura 2.11(c)) e a segmentação instanciada (Figura 2.11(d)) é que na segmentação instanciada o modelo consegue diferenciar objetos que possuem a mesma classe.

A popularização do uso de CNNs para as tarefas citadas na Figura 2.11 aconteceu em 2012 quando a AlexNet (Krizhevsky et al., 2012) ganhou a competição *ImageNet Large Scale Visual Recognition Challenge 2012* (ILSVRC-2012) utilizando redes neurais convolucionais. A competição avaliou arquiteturas para classificação de imagens na ImageNet (Deng et al., 2009) que reúne 1,2 milhões de imagens de alta-resolução separadas em 1000 diferentes classes.

As Figuras 2.12 e 2.13, apresentam a evolução da acurácia das arquiteturas durante os anos de 2011 a 2022 na ImageNet. A Figura 2.12 apresenta a porcentagem Top-5 que considera que o modelo acertou se a classe estiver presente nas cinco classes prováveis e a Figura 2.13 apresenta a porcentagem de acerto Top-1 que avalia somente a classe com maior probabilidade.

Nas próximas duas seções serão apresentadas as arquiteturas referências para as tarefas principais do presente trabalho, classificação e segmentação semântica, respectivamente.

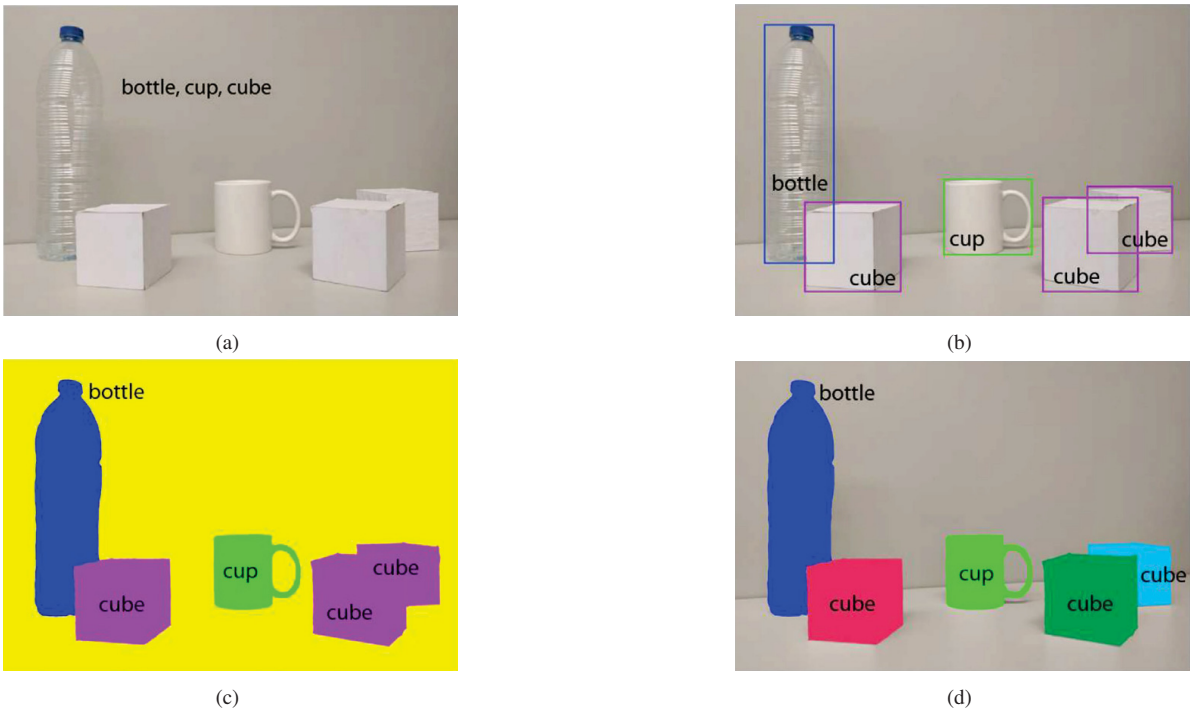


Figura 2.11: Evolução das tarefas em visão computacional - (a) classificação (b) localização de objetos (c) segmentação semântica (d) segmentação instanciada. Fonte: (Garcia-Garcia et al., 2018)

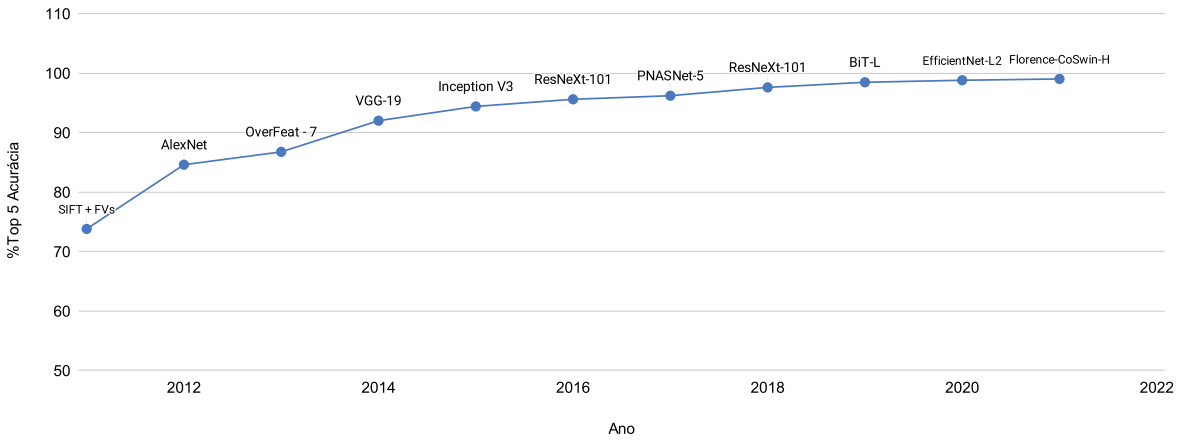


Figura 2.12: Evolução da acurácia Top-5 na ImageNet entre os anos de 2011 a 2022

2.8.1.1 Arquiteturas para classificação

A AlexNet (Krizhevsky et al., 2012), ilustrada na Figura 2.14 foi campeã do ILSVRC-2012. A arquitetura foi a primeira a obter resultados satisfatórios na competição utilizando redes neurais convolucionais. No projeto de 2012 a arquitetura foi treinada em duas GPUs devido à limitação de memória das placas na época. O modelo também popularizou o uso das camadas de ativação do tipo ReLU. Em comparação, na avaliação Top-5 a AlexNet atingiu 15,3% de erro e o segundo colocado obteve 26,2%, expressando o potencial das CNNs.

A arquitetura VGG (Simonyan e Zisserman, 2014) criada pelo grupo *Visual Geometry Group* da Universidade de Oxford, vice colocada no ILSVRC-2014, atingiu 92,7% no teste Top-5 ficando 0,6% atrás da GoogLeNet (Szegedy et al., 2014). A contribuição principal do projeto

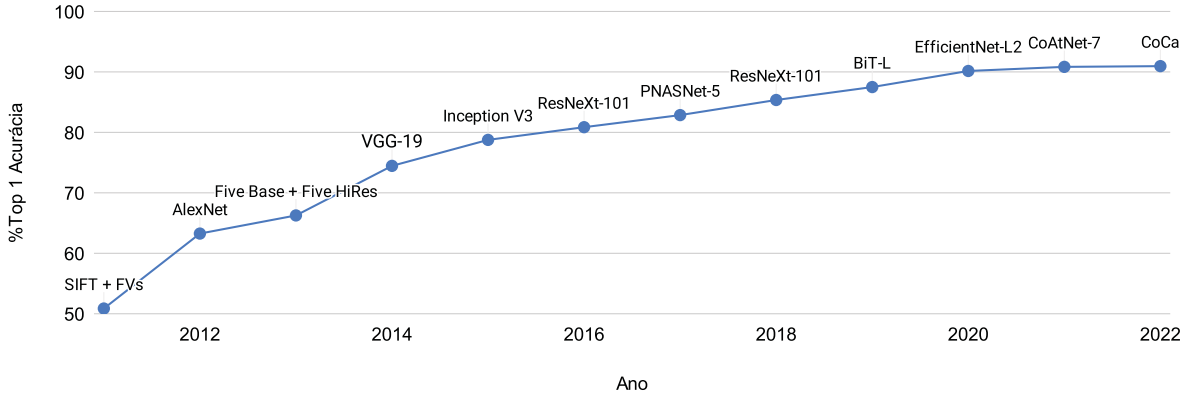


Figura 2.13: Evolução da acurácia Top-1 na ImageNet entre os anos de 2011 a 2022

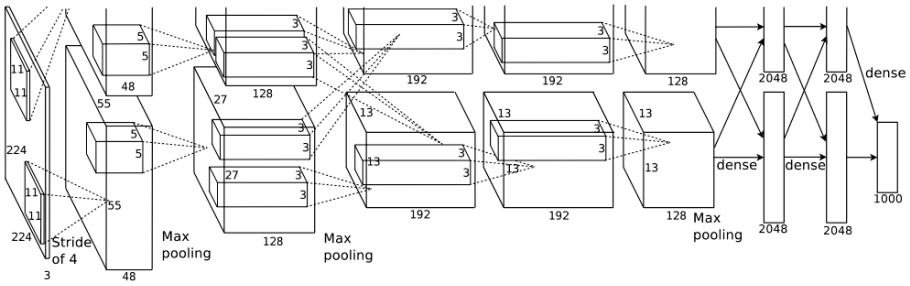


Figura 2.14: AlexNet. Fonte: (Krizhevsky et al., 2012)

foi a utilização de filtros convolucionais menores (3 × 3) em relação aos utilizados na AlexNet, possibilitando assim uma rede neural mais profunda com 19 camadas.

A GoogLeNet (Szegedy et al., 2014), campeã da competição ILSVRC-2014 com 93,3% na avaliação Top-5 introduziu o módulo *Inception* (Figura 2.15) constituído de várias operações de convolução com diferentes tamanhos de filtros sendo concatenados no final. Esses módulos reduzem o número de parâmetros presentes na arquitetura, possibilitando assim uma arquitetura de 22 camadas e com 12 vezes menos parâmetros que a AlexNet.

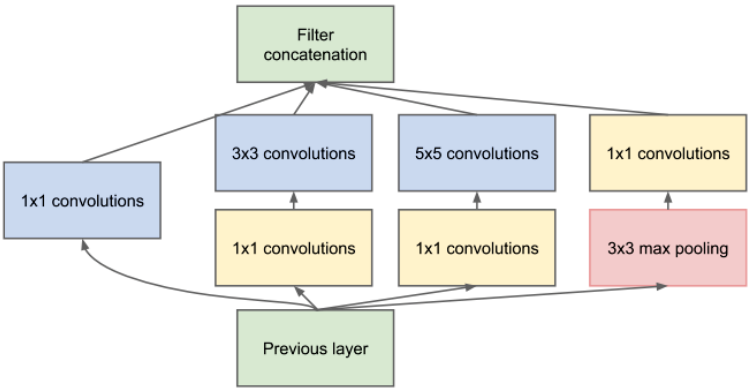


Figura 2.15: Módulo *Inception* presente na GoogLeNet. Fonte: (Szegedy et al., 2014)

Desenvolvida pela Microsoft, a ResNet (He et al., 2015) possui blocos residuais (Figura 2.16) que passa as informações de camadas superiores (x) para camadas inferiores ($F(x)$) para propagar gradiente por toda a rede neural evitando o problema de desaparecimento do gradiente

que impossibilita o treinamento de arquiteturas a partir de um determinado número de camadas (Glorot e Bengio, 2010).

Com a implementação dos blocos residuais e a adição de camadas de *batch normalization* (Ioffe e Szegedy, 2015), a ResNet atingiu a marca de 152 camadas em sua arquitetura. O modelo foi o vencedor do ILSVRC-2015 com uma porcentagem de acerto de 96,43% no teste Top-5.

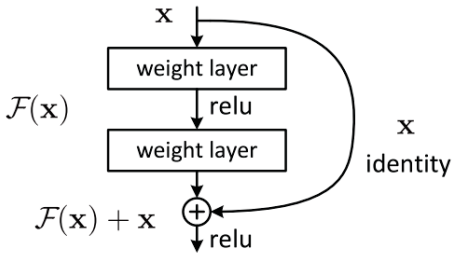


Figura 2.16: Bloco residual presente na ResNet. Fonte: (He et al., 2015)

2.8.1.2 Arquiteturas para segmentação semântica

As arquiteturas para segmentação semântica popularmente utilizam a base de dados Pascal VOC 2012 (Everingham et al., 2012) para realizar seus *bechmarks*. O Pascal VOC 2012 possui 21 classes e constituído de 1464 de treino e 1449 imagens de validação, as informações das imagens de teste são mantidas privadas. Os *bechmarks* realizados em segmentação semântica utilizam a métrica *Mean Intersection-Over-Union* (mIoU). A evolução da mIoU na base de teste do Pascal VOC 2012 foi apresentada na Figura 2.17.

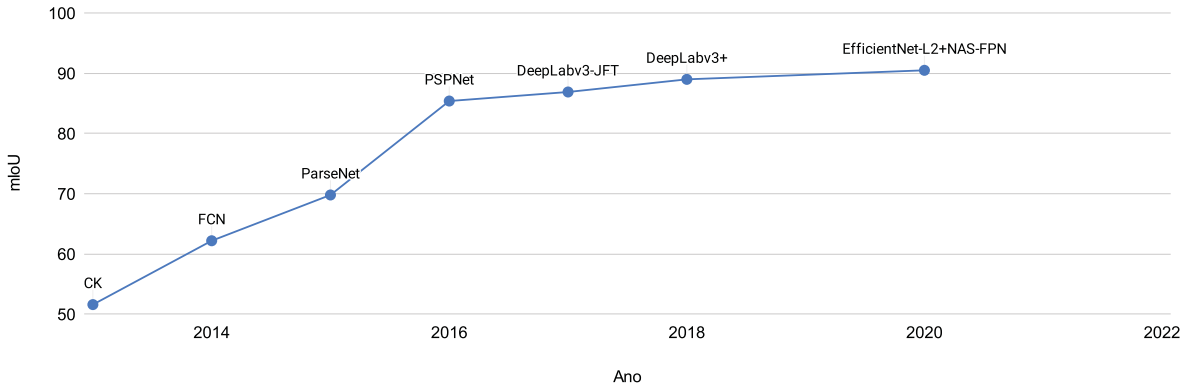


Figura 2.17: Evolução do mIoU na Pascal VOC 2012 entre os anos de 2013 a 2022

A arquitetura *Fully Convolutional Network* (FCN) (Long et al., 2014), foi desenvolvida transformando as camadas densas das arquiteturas de classificação AlexNet (Krizhevsky et al., 2012), VGG (Simonyan e Zisserman, 2014) e GoogLeNet (Szegedy et al., 2014) em camadas convulsionais.

A arquitetura atingiu 62,2 de mIoU na base PASCAL VOC 2012 com o modelo utilizando a VGG como *backbone*. Como demonstrado na Figura 2.17 o impacto do desenvolvimento da FCN na área de segmentação semântica foi similar ao da AlexNet para classificação em 2012 (Figura 2.12).

Com base no modelo proposto pela FCN, de uma arquitetura somente com camadas convolucionais, a Segnet (Badrinarayanan et al., 2017) foi desenvolvida visando melhorar a

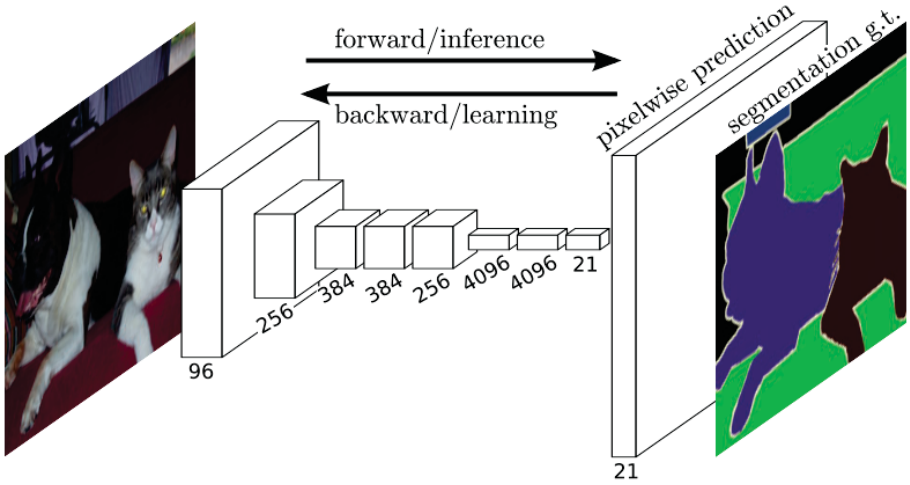


Figura 2.18: FCN. Fonte: (Long et al., 2014)

parte do *upsampling* do *feature map* trocando o modelo que possuía somente algumas camadas convolucionais para uma arquitetura simétrica (Figura 2.19) com duas regiões, o *encoder* que reduz a dimensão gerando o *feature map* e um *decoder* que realiza o *upsampling* do *feature map* para retornar o mapa de segmentação. Na SegNet em sua operação de *upsampling* utiliza a localização dos índices da operação de *max-pooling* (Figura 2.20).

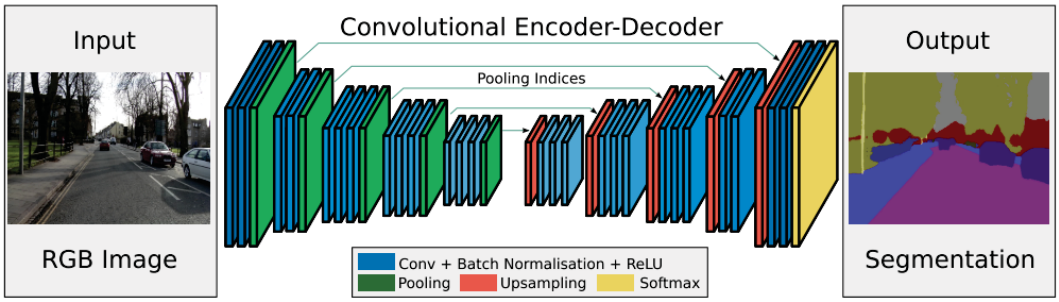


Figura 2.19: SegNet. Fonte: (Badrinarayanan et al., 2017)

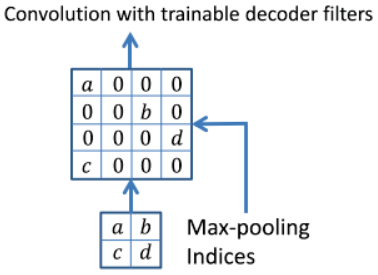


Figura 2.20: *Upsampling* utilizando os *max pooling* índices na SegNet. Fonte: (Badrinarayanan et al., 2017)

A U-net (Ronneberger et al., 2015) foi uma arquitetura criada para segmentação semântica binária de imagens biomédicas. Teve como a sua maior contribuição a utilização de *skip-connections* (Figura 2.21) que concatena o *feature map* do *encoder* no *decoder* assim transferindo informação para operação de *upsampling*. O modelo *encoder-decoder* utilizando *skip-connections* da U-net foi aperfeiçoado em outras arquiteturas como ExFuse (Zhang et al., 2018), DeepLab V3+ (Chen et al., 2018) e RefineNet (Lin et al., 2016).

Em 2017, a arquitetura foi modificada adicionando módulos residuais como a ResNet (He et al., 2015) para extração de rodovias em imagens áreas (Zhang et al., 2017). Atualmente, o *framework* Segmentation Models (Yakubovskiy, 2019) possibilita a utilização da U-net com diferentes *encoders*, como a VGG (Simonyan e Zisserman, 2014), ResNet (He et al., 2015), EfficientNet (Tan e Le, 2019).

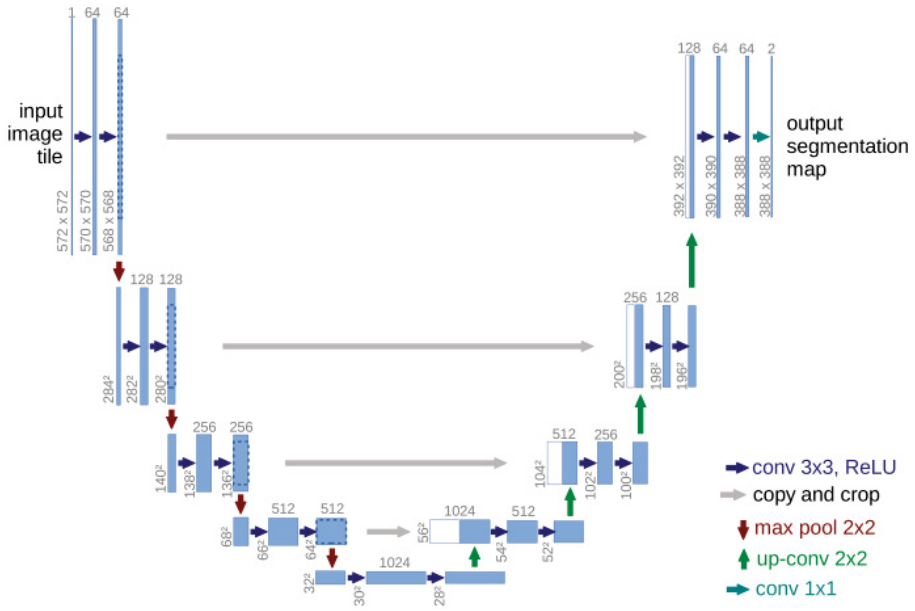


Figura 2.21: U-net. Fonte: (Ronneberger et al., 2015)

Pyramid Scene Parsing Network (PSPNet) (Zhao et al., 2016) (Figura 2.23) atingiu 85,4 de mIoU no Pascal VOC 2012. A arquitetura utilizou a Resnet-101 (He et al., 2015), como *backbone*, entretanto foram adicionadas camadas com convolução dilatada (Figura 2.22).

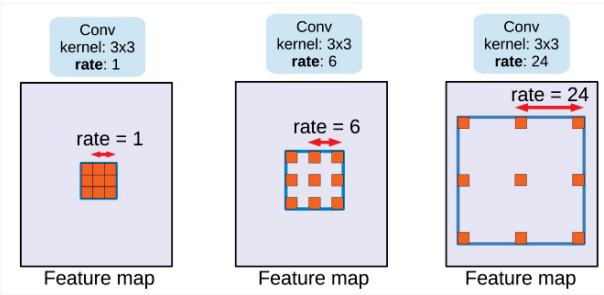


Figura 2.22: Comparação entre a convolução padrão e a dilatada. O *rate* indica o fator de dilatação da convolução. *Rate* = 1 é a convolução padrão. Fonte: (Chen et al., 2016)

A rede possui um módulo de *pyramid pooling* que opera sobre o *feature map* com filtros com diferentes tamanhos que, após suas operações, sofrem *upsampling* sendo concatenados com o *feature map* original.

Em 2018 a Google lançou a Deeplab V3+ (Chen et al., 2018) (Figura 2.24) evoluindo a versão da DeepLab V3 (Chen et al., 2017) em relação ao seu *decoder* e utilizando a arquitetura Xception (Chollet, 2016), modificada pela equipe *Microsoft Research Asia* (MSRA) (Dai et al., 2017), como *encoder*. A arquitetura atingiu 89.0 de mIoU no Pascal VOC 2012.

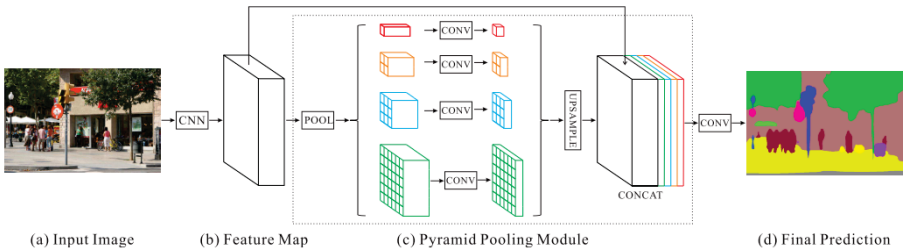


Figura 2.23: PSPNet. Fonte: (Zhao et al., 2016)

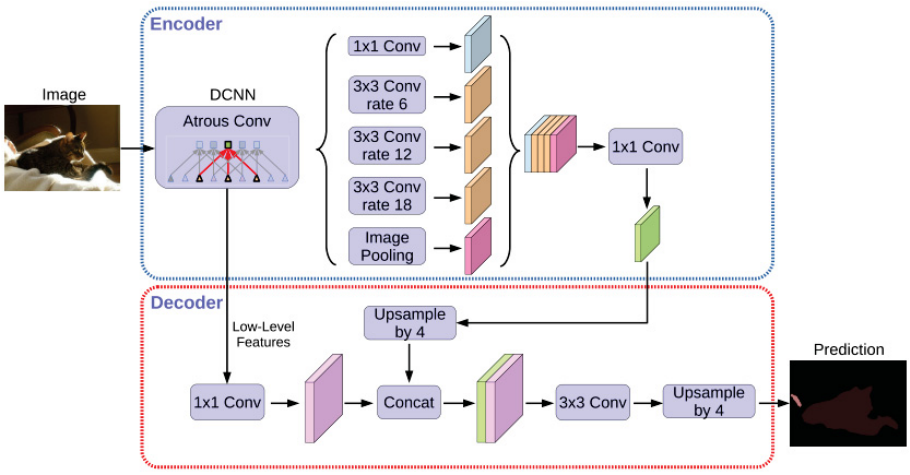


Figura 2.24: DeepLab V3+. Fonte: (Chen et al., 2018)

2.8.2 Interpretabilidade

Atualmente, com os avanços das CNNs suas estruturas foram ficando cada vez mais complexas e, em contrapartida, a interpretabilidade dos resultados foi ficando cada vez menor. A utilização somente de métricas para avaliação das predições não é o suficiente para compreender como os modelos estão realizando suas predições.

Outro ponto é que as CNNs ainda são modelos vulneráveis a *adversarial attack* (Goodfellow et al. (2015), Madry et al. (2018)). O *adversarial attack* constitui em modificar a entrada do modelo adicionando pequenos ruídos constituídos de sinais dos elementos do gradiente da função de custo da entrada.

Na Figura 2.25 apresenta um exemplo de um *adversarial attack* que ao combinar a imagem de um panda com um ruído proveniente do gradiente da sua função de custo resulta em uma troca significativa da predição do modelo. O modelo testado com a alteração da imagem prediz a classe *gibbon* com 99,3% de confiança. As mudanças na imagem são imperceptíveis para o ser humano, entretanto para o modelo essas mudanças alteram completamente a resposta da rede neural.

Para ajudar na interpretabilidade foram desenvolvidos algoritmos que auxiliam nessa tarefa. O algoritmo *Local Interpretable Model-agnostic Explanations* (LIME) subdivide a imagem em super-píxeis e realiza perturbações na imagem. Posteriormente, o algoritmo avalia os resultados utilizando *Least Angle Regression* (Lasso) (Efron et al., 2004) para selecionar quais super-pixeis são os que explicam a predição do modelo.

Ribeiro et al. (2016) apresentaram resultados do algoritmo LIME utilizando a GoogleNet (Szegedy et al., 2014) pré-treinada na ImageNet. A imagem utilizada apresentada na Figura 2.26(a) é uma junção de várias classes presentes na ImageNet. Segundo o modelo, a imagem é

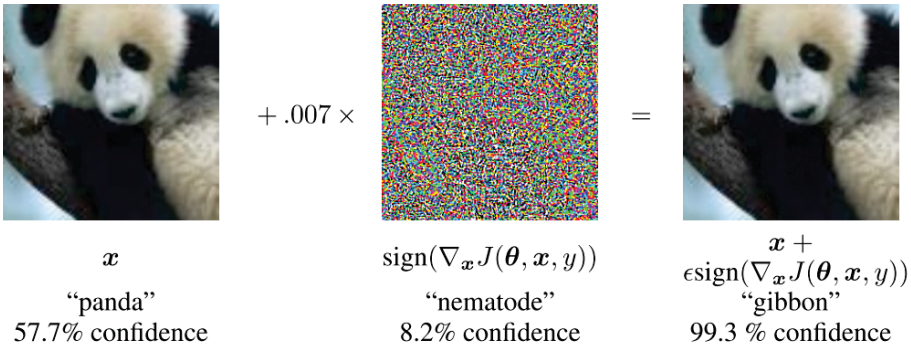


Figura 2.25: Demonstração de um *adversarial attack* para uma imagem de um panda utilizando o sinal do gradiente da sua função de custo. Fonte: (Goodfellow et al., 2015)

da classe *guitarra* com probabilidade de 0,32, em seguida da classe *violão* com probabilidade de 0,24 e da classe *labrador* com probabilidade de 0,21.

Em primeiro momento a predição do modelo não é esperada, entretanto, analisando os resultados do algoritmo LIME (Figuras 2.26(b), 2.26(c) e 2.26(d)) observa-se que o cabo do violão é parecido com um cabo de uma guitarra e que o modelo reconhece bem as regiões que pertencem às outras classes *violão* e *labrador*.

(a) (b) (c) (d)

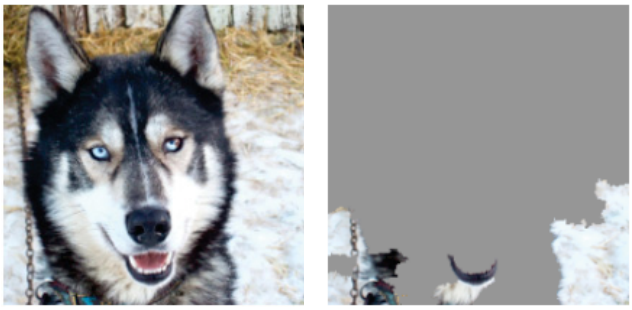
Figura 2.26: Demonstração dos resultados do algoritmo LIME utilizando a GoogleNet (Szegedy et al., 2014), pré-treinada na ImageNet - (a) imagem original (b) classe *guitarra* (c) classe *violão* (d) classe *labrador*. Fonte: (Ribeiro et al., 2016)

Os autores do algoritmo LIME também avaliaram os resultados em uma base de dados de 60 imagens que comparam *Husky* e *Lobo* e utilizaram a GoogleNet como extrator de característica e para a classificação utilizaram regressão logística. Utilizando uma imagem classificada incorretamente como *lobo* (Figura 2.27(a)) e o algoritmo LIME os autores observaram que o modelo para a classe *lobo* se baseava na neve encontrada na imagem para realizar suas predições e não nas características presentes no animal (Figura 2.27(b)).

Outro método para interpretação das predições dos modelos utiliza os valores de Shapley. Os valores de Shapley demonstram o impacto de uma determinada característica no valor esperado para o seu resultado utilizando teoria dos jogos. Todavia, o cálculo dos valores de Shapely necessita de uma alto custo computacional.

Lundberg e Lee (2017) desenvolveram o algoritmo *Shapley Additive exPlanations* (SHAP) que utiliza regressão linear local ponderada para aproximar os valores de Shapley. Os autores também desenvolveram variações específicas do SHAP para os modelos de aprendizagem de máquina baseados em árvores, *deep learning* e lineares para reduzir o custo computacional.

A Figura 2.28 apresenta os resultados dos valores de SHAP para 4 imagens e para 10 classes (dígitos 0-9). Os píxeis marcados em vermelho representam valores positivos de SHAP, mostrando que essas regiões impactam positivamente a saída do modelo para aquela classe. Os pontos azuis representam valores negativos de SHAP, informando que essas regiões diminuem a



(a) (b)

Figura 2.27: Explicação de uma predição incorreta utilizando LIME - (a) imagem original (b) explicação da classificação como *lobo*. Fonte: (Ribeiro et al., 2016)

saída do modelo para aquela classe. Nota-se que para a imagem do ‘zero’ a parte vazia no centro é importante, enquanto para a imagem do ‘quatro’, a falta da conexão na parte superior diferenciá ela do dígito ‘nove’.

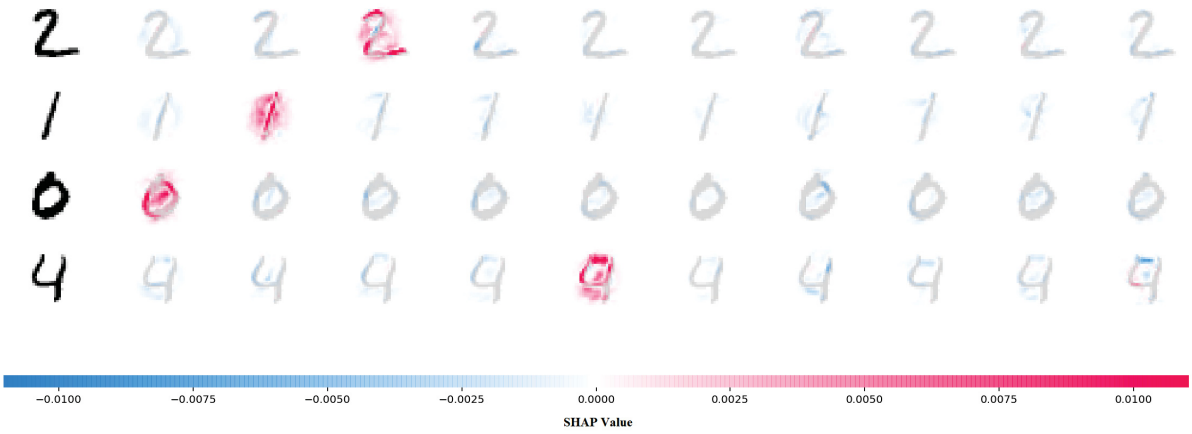


Figura 2.28: Exemplo do algoritmo SHAP para classificação de imagens contendo dígitos. Os pontos em azul e vermelho representam valores negativos e positivos dos valores SHAP, respectivamente. Fonte: (SHAP, 2021)

2.9 MÉTRICAS DE AVALIAÇÃO

Essa seção apresenta as principais métricas utilizadas para avaliação dos modelos para as tarefas de classificação (Seção 2.9.1), segmentação semântica (Seção 2.9.2) e localização de objetos (Seção 2.9.3).

2.9.1 Métricas para tarefa de classificação

Para a classificação utiliza-se popularmente a análise da matriz de confusão e as métricas de acurácia, *F1 score* (F1) e o *Matthews Correlation Coefficient* (MCC) para avaliar o desempenho dos modelos.

A matriz de confusão (Tabela 2.1) apresenta os resultados da classificação na forma de uma matriz, onde as linhas são as classes reais e as colunas são as classes preditas. No interior da matriz, encontra-se os valores de *True Positive* (TP), *False Positive* (FP), *True Negative* (TN) e *False Negative* (FN).

A acurácia (Equação 2.19) é uma métrica que avalia a porcentagem de acertos positivos e negativos do modelo (TP e TN). É a métrica mais popular para classificação, entretanto para

Tabela 2.1: Matriz de confusão

		Valor Predito	
		Positivo	Negativo
Real	Positivo	TP	FN
	Negativo	FP	TN

problemas binários desbalanceados não é recomendado seu uso. Em problemas desbalanceados, classificadores que possuem uma tendência de predizer a classe dominante são pouco penalizados pela acurácia.

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.19}$$

O F1 (Equação 2.22) é a média harmônica entre o *recall* (Equação 2.20) e a precisão (Equação 2.21). Devido à utilização da média harmônica para a obtenção de melhores resultados os valores de *recall* e precisão devem ser próximos.

$$Recall = \frac{TP}{TP + FN} \tag{2.20}$$

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{2.21}$$

$$F1 = 2\,\frac{\text{Precisão } x\ Recall}{\text{Precisão} + Recall} \tag{2.22}$$

O *recall* avalia a capacidade do classificador encontrar todos os dados positivos, sendo uma métrica relevante, por exemplo, em classificações médicas. A precisão avalia competência do classificador em não prever FP, utiliza-se essa métrica em casos que a predição de FP gera um alto custo, por exemplo, em detecção de e-mails *spam*.

O F1 é recomendado para classificações binárias com dados desbalanceados, pois classificadores que predizem a classe dominante não obtém resultados desejados no F1 devido ao baixo resultado do recall ou da precisão.

Para problemas categóricos, bibliotecas populares para aprendizagem de máquinas como o Sklearn (Buitinck et al., 2013) utiliza-se o F1 macro que realiza a média dos valores de F1 (Equação 2.23) para cada classe do problema, onde N é o número de classe do problema.

$$F1\ (\text{Macro}) = \frac{1}{N}\sum_{N} F1_N \tag{2.23}$$

Para problemas categóricos, o F1 (Macro) possui o problema de priorizar as classes minoritárias. Beijbom et al. (2012) em seu trabalho de classificação de recifes de corais mesmo com o desbalanceamento das classes julgou que a utilização da acurácia era uma melhor opção em relação ao F1, pois não existe diferença de peso entre as classes mesmo que elas sejam majoritárias ou minoritárias no problema. Na competição da ImageNet como demonstrado na Figura 2.13 utiliza-se também a acurácia para avaliar o desempenho dos classificadores.

O MCC (Equação 2.24) avalia os resultados em um intervalo entre -1 e 1. $MCC = 1$ representa uma predição perfeita, $MCC = 0$ uma predição que não é melhor que uma predição aleatória e $MCC = -1$ uma predição inversa. Conforme o autor Chicco e Jurman (2020), o MCC para classificações binárias é mais confiável que o F1, pois ele somente produz altos resultados

se o classificador encontra bons resultados nas 4 categorias da matriz de confusão (TP, FN, TN, FP).

$$MCC = \frac{TP\ x\ TN - FP\ x\ FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{2.24}$$

2.9.2 Métricas para tarefa de segmentação semântica

As métricas de segmentação semântica podem ser interpretadas por uma análise gráfica. A sua matriz de confusão pode ser representada como na Figura 2.29. A área quadrada representa o *ground truth*, a área circular é a área predita. A Figura 2.29 representa um problema binário, então o restante da imagem é o plano de fundo, onde se encontra o TN. As métricas populares para segmentação semântica são: acurácia por píxel e o mIoU.

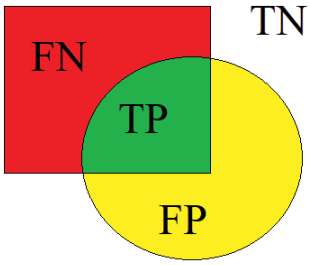


Figura 2.29: Representação visual do TP, TN, FP e FN para a tarefa de segmentação semântica

A métrica de acurácia por píxel segue a mesma fórmula da acurácia da tarefa de classificação (Equação 2.19). Utiliza-se o nome acurácia por píxel, pois a métrica avalia os píxeis presentes na predição do modelo.

No caso da segmentação normalmente a classe plano de fundo possui um número de píxel relativamente maior que as outras classes. Portanto, a utilização da métrica pode resultar em avaliações superestimadas.

A Figura 2.30 apresenta visualmente a métrica *Intersection-Over-Union* (IoU) que é a área de intersecção entre o *ground truth* e a predição do modelo dividido pela área da união entre os dois conjuntos. Essa é a principal métrica utilizada em segmentação semântica.

Para problemas binários ou categóricos é calculado o mIoU que realiza a média dos IoU para cada classe k (Equação 2.25).

$$mIoU = \frac{1}{k}\sum_{k}\frac{TP_k}{FP_k + FN_k + TP_k} \tag{2.25}$$

2.9.3 Métricas para tarefa de localização de objetos

Para a tarefa de localização de objetos a métrica mais popular utilizada em competições nas bases PASCAL VOC (Everingham et al., 2012) e COCO (Lin et al., 2014) é a *Average Precison* α (APα). A métrica é calculada a partir da área abaixo da curva de precisão-recall (Figura 2.31) avaliada em um limiar α de IoU. Portanto, AP50 e AP95 identifica que a métrica é calculada nos limiares de $IoU = 0,5$ e $IoU = 0,95$, respectivamente. A Equação 2.26 apresenta a definição da métrica APα, onde p é a precisão e r é o *recall*.

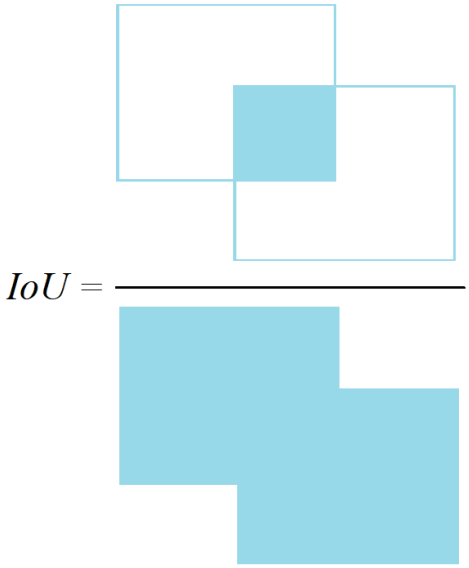


Figura 2.30: Representação visual do IoU

$$AP\alpha = \int_0^1 p(r)\, dr \tag{2.26}$$

Como a métrica mIoU, a métrica *Mean Average Precison* α (mAPα) somente é a média da métrica APα para todas a classes n definidas para o problema (Equação 2.27).

$$mAP\alpha = \frac{1}{n} \sum_n AP\alpha_i \tag{2.27}$$

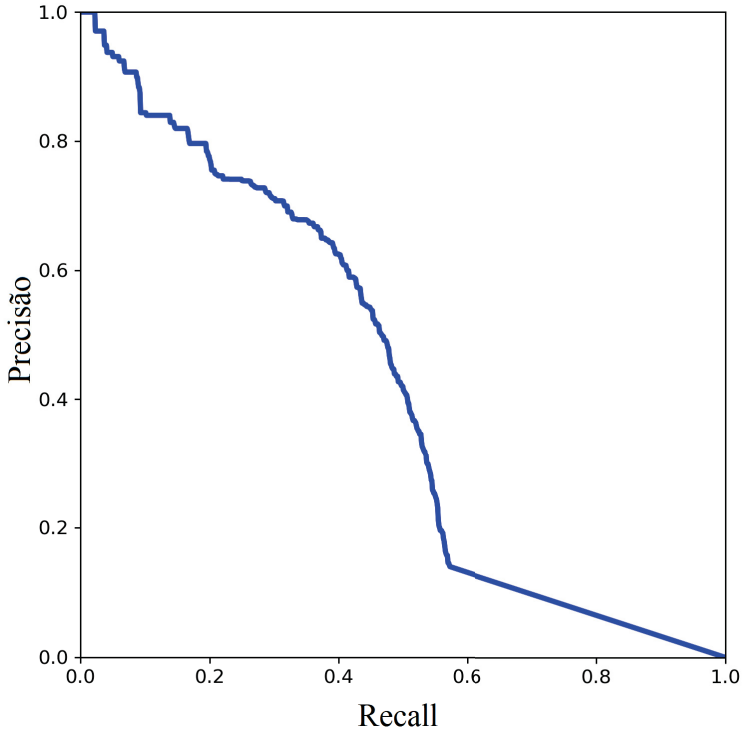


Figura 2.31: Exemplo da curva de precisão-recall

3 TRABALHOS CORRELATOS

Esse capítulo apresenta o estado da arte sobre trabalhos relacionados a classificação e segmentação semântica de recifes de corais utilizando aprendizagem de máquina. A Seção 3.1 descreve as principais bases de dados públicas. As Seções 3.2 e 3.3 apresentam os principais trabalhos que utilizam aprendizagem de máquina para classificar e segmentar semanticamente recifes de corais, respectivamente. Por fim, a Seção 3.4 realiza uma análise crítica do estado da arte.

3.1 BASE DE DADOS

As principais base de dados públicas disponíveis para recifes de corais são:

- *Moorea Labeled Corals* (MLC)[1]
- *Pacific Labeled Corals* (PLC)[2]
- *Eilat Dataset* (EILAT)[3]
- *Eilat 2 Dataset* (EILAT 2)[3]
- *Rosenstiel School of Marine and Atmospheric Sciences* (RSMAS)[3]

A base de dados MLC é uma sub-base da *Moorea Coral Reef-Long Term Ecological Research* (MLC-LTER), introduzida no trabalho de Beijbom et al. (2012). A MLC contém 400.000 anotações realizadas por especialistas em 2055 imagens.

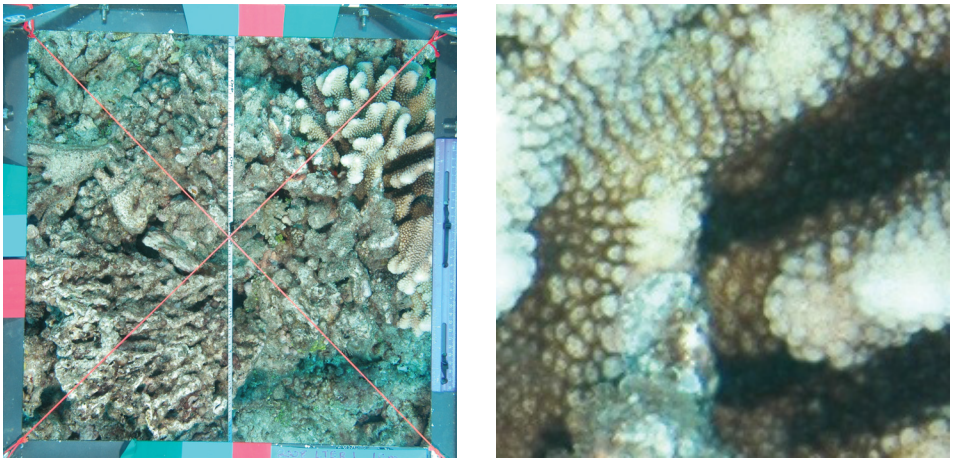
As anotações são pixels marcados informando que naquela região existe um objeto correspondente a uma determinada classe. Normalmente se extrai sub-imagens centradas na anotação com uma resolução predefinida para se avaliar o objeto.

A Figura 3.1 apresenta duas imagens, a Figura 3.1(a) é a imagem completa com resolução de 1952 × 1952 e a Figura 3.1(b) uma sub-imagem centrada em uma anotação com resolução de 224 × 224 da classe *Pocillopora*. Para a Figura 3.1(a) existem 200 anotações. A Tabela 3.1 apresenta 5 exemplos de anotações da Figura 3.1(a).

Tabela 3.1: Cinco anotações presentes na base MLC para a Figura 3.1(a)

Rótulo	**Linha**	**Coluna**
CCA	134	730
Turf algae	1441	889
Pocillopora	725	1642
CCA	1597	391
Turf algae	1250	573

[1] http://vision.ucsd.edu/content/moorea-labeled-corals
[2] http://vision.ucsd.edu/content/pacific-labeled-corals
[3] https://data.mendeley.com/datasets/86y667257h/2



(a) (b)

Figura 3.1: Exemplo de utilização da base MLC - (a) imagem completa (b) sub-imagem

As imagens foram extraídas da ilha Moorea na Polinésia Francesa entre os anos de 2008 a 2010 e contém 9 classes divididas em dois grupos: não-coral e coral. Na base MLC os corais são classificados de acordo com seu gênero. As classes são:

- Não-Coral: *Crustose Coralline Algae* (CCA), *Turf algae*, *Macroalgae* e *Sand*
- Coral: *Acropora*, *Pavona*, *Montipora*, *Pocillopora* e *Porites*

A base de dados PLC, introduzida no trabalho de Beijbom et al. (2015) para estimar a cobertura das classes na imagem, é um agregado de 5090 imagens de 4 regiões do Pacífico: Moorea (Polinésia Francesa), Ilhas da Linha do norte, *Nanwan Bay* (Taiwan) e *Heron Reef* (Austrália). Foi desenvolvida pelo mesmo grupo de pesquisa que desenvolveu a base MLC.

A PLC contém 251,988 anotações realizadas por especialistas e contém 20 classes: *Acropora*, *Favia*, *Favites*, *Montipora*, *Pavona*, *Platygyra*, *Pocillopora*, *Porites*, *Other scleractinians*, *Millepora*, *Sponges*, *Soft Coral*, CCA, *Turf*, *Macroalgae*, *Sand*, *Bare Substrate*, *Transect hardware*, *Unclear*, *All other*. A PLC classifica os corais pelo gênero como a MLC. Um ponto a se destacar é a falta de trabalhos que utilizam essa base para realizar a classificação das imagens.

A base EILAT contém 1123 imagens com resolução de 64 x 64 pixeis, os corais são do golfo de Eilat no mar vermelho. A EILAT contém 8 classes: *sand*, *urchin*, *branches type I*, *brain coral*, *favid coral*, *branches type II*, *dead coral* e *branches type III*. A base EILAT 2 contém 303 imagens com resolução de 128 × 128 pixels separados em 5 classes: *sand*, *urchin*, *branching coral*, *brain coral* e *favid coral*.

A base RSMAS foi coletada por mergulhadores da *Rosenstiel School of Marine and Atmospheric Sciences* da Universidade de Miami. Ela contém 766 imagens com resolução de 256 × 256 pixels e contém 14 classes: *Acropora cervicornis*, *Acropora palmata*, *Colpophyllia natans*, *Diadema antillarum*, *Diploria strigosa*, *Gorgonians*, *Millepora alcicornis*, *Montastraea cavernosa*, *Meandrina meandrites*, *Montipora spp.*, *Palythoas palythoa*, *Sponge fungus*, *Siderastrea siderea* e *tunicates*. No caso da RSMAS os corais são classificados segundo a sua espécie. Os corais presentes na RSMAS possuem uma relação próxima com os corais brasileiros por estarem localizados na região do Caribe, em comparação aos do Pacífico que embasam as demais bases.

As três bases (EILAT, EILAT 2 e RSMAS) podem ser obtidas através do trabalho de Shihavuddin (2017). A Tabela 3.2 apresenta a comparação entre as bases públicas de recifes de corais.

Tabela 3.2: Levantamento sobre as bases de dados públicas de recifes de corais

Nome	Anos de coleta	Núm. de imagens	Núm. de anotações	Núm. de classes	Resolução
MLC	2008, 2009, 2010	2055	400.000	9	-
PLC	2005, 2007, 2008, 2012	5090	251.988	20	-
EILAT	-	1123	-	8	64 x 64
EILAT 2	-	303	-	5	128 x 128
RSMAS	-	766	-	14	256 x 256

Recentemente, foi lançado a CoralNet que é um repositório e uma plataforma de colaboração para imagens de organismos bentônicos. O site possui uma rede neural (Williams et al., 2019) que habilita a anotação semi ou totalmente automática das imagens. Com as ferramentas disponíveis no site e a implementação de uma anotação automática espera-se que a CoralNet auxilie várias pesquisas relacionadas a organismos bentônicos mundialmente.

3.2 CLASSIFICAÇÃO DE RECIFES DE CORAIS UTILIZANDO APRENDIZAGEM DE MÁQUINA

Beijbom et al. (2012) utilizaram em seu trabalho a base de dados MLC. O trabalho utilizou um filtro de resposta máxima com o classificador SVM para classificar as imagens. Os experimentos foram realizados de 3 formas: somente com a base de 2008 (exp-1), com a base de 2008 para treino e base de 2009 para teste (exp-2) e a base de 2008 e 2009 para treino e a base de 2010 para teste (exp-3). Os resultados reportados para os experimentos 1, 2 e 3 encontram-se na Tabela 3.3.

Tabela 3.3: Resultados de acurácia para os 3 experimentos realizados por Beijbom et al. (2012)

	Exp-1	Exp-2	Exp-3
Beijbom et al. (2012)	0,740	0,670	0,830

Shihavuddin et al. (2013) utilizaram as bases MLC (2008), EILAT e RSMAS para realizar a classificação dos corais. Na base MLC (2008) retirou-se sub-imagens de 312 × 312 pixels das anotações. O modelo proposto é em uma junção de diversas etapas para se realizar a classificação e o treinamento dos modelos. Para a extração de característica utilizaram-se os seguintes algoritmos:

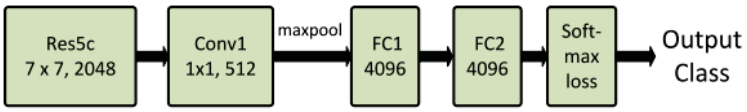
- *Completed Local Binary Pattern* (CLBP)
- *Grey Level Co-occurence Matrix* (GLCM)
- Filtro Gabor
- Histograma do ângulo oponente e do canal de matiz

Os principais resultados desse correlato encontram-se na Tabela 3.4, utilizou-se o classificador *K-Nearest Neighbors* (KNN) para a EILAT e a RSMAS e o *Probability Density Weighted Mean Distance* (PDWMD) para a MLC 2008. Os resultados para a MLC 2008 foram superiores aos encontrados por Beijbom et al. (2012) e são os melhores encontrados na literatura atualmente, apesar do número de etapas para se realizar a classificação.

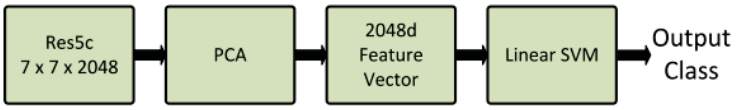
Tabela 3.4: Resultados de acurácia encontrados por Shihavuddin et al. (2013) para as bases: EILAT, RSMAS e MLC 2008

	EILAT	**RSMAS**	**MLC 2008**
Shihavuddin et al. (2013)	0,969	0,965	0,855

Mahmood et al. (2016) apresentaram um modelo utilizando a ResNet (He et al., 2015) como extrator de característica e utilizou dois modelos para realizar a classificação. O primeiro é uma CNN e o outro é um *Principal Component Analysis-Support Vector Machine* (PCA-SVM). Os autores utilizaram a base MLC (2008) para avaliar o modelo com sub-imagens de 224 × 224 pixels. A Tabela 3.5 apresenta os resultados encontrados, observa-se que os resultados obtidos foram inferiores aos encontrados por Shihavuddin et al. (2013).



(a)



(b)

Figura 3.2: Resfeats - (a) sCNN e (b) PCA-SVM. Fonte: (Mahmood et al., 2016)

Tabela 3.5: Resultados de acurácia encontrados por Mahmood et al. (2016) para a base MLC 2008 e comparação com os resultados encontrados por Shihavuddin et al. (2013)

	MLC 2008
ResFeats-50 + sCNN	0,788
ResFeats-152 + sCNN	0,800
ResFeats-152 + PCA-SVM	0,808
Shihavuddin et al. (2013)	0,855

Xu et al. (2019) avaliaram o uso da DenseNet-121 na base MLC com sub-imagens de 224 × 224. O autor reproduziu os 3 experimentos realizados por Beijbom et al. (2012). A Tabela 3.6 apresenta os resultados encontrados e a comparação com os resultados encontrados por Beijbom et al. (2012) e Shihavuddin et al. (2013).

Tabela 3.6: Resultados de acurácia encontrados por Xu et al. (2019) na base MLC e comparação com os resultados encontrados por Beijbom et al. (2012) e Shihavuddin et al. (2013)

	Exp-1	**Exp-2**	**Exp-3**
Beijbom et al. (2012)	0,740	0,670	0,830
Xu et al. (2019)	0,845	0,742	0,865
Shihavuddin et al. (2013)	0,855	-	-

Observa-se que os resultados foram melhores que os encontrados por Beijbom et al. (2012). Entretanto, a utilização de CNNs pré-treinadas na literatura é inferior aos resultados encontrados por Shihavuddin et al. (2013) para a base MLC (2008).

Gómez-Ríos et al. (2019) avaliaram a utilização de CNNs para classificar corais em duas bases de dados EILAT e RSMAS. Os autores utilizaram os modelos Inception v2 (Ioffe e Szegedy, 2015), ResNet (He et al., 2015), DenseNet (Huang et al., 2016). O trabalho ainda testou técnicas de *data augmentation* (zoom, translação, giro e rotação) para melhorar o desempenho dos modelos.

Os melhores resultados reportados utilizaram a ResNet-50 com *data augmentation*. O modelo obteve nas bases de dados EILAT e RSMAS 0,9803 e 0,9863 de acurácia, respectivamente.

Lumini et al. (2019) também avaliaram a utilização de CNNs nas bases de dados EILAT e RSMAS como Gómez-Ríos et al. (2019). No trabalho não utilizou-se técnicas de *data augmentation* e testou-se *ensembles* entre os classificadores para melhorar o desempenho. Os modelos testados foram AlexNet (Krizhevsky et al., 2012), GoogleNet (Szegedy et al., 2014), InceptionV3 (Szegedy et al., 2015), VGG (Simonyan e Zisserman, 2014), DenseNet (Huang et al., 2016), NasNet (Zoph et al., 2017) e MobileNetV2 (Sandler et al., 2018).

O melhor resultado reportado foi do *ensemble* denominado SFFS que possuía 11 classificadores. O *ensemble* (SFFS) obteve nas bases de dados EILAT e RSMAS 0,989 e 0,992 de acurácia, respectivamente. A Tabela 3.7 apresenta a comparação dos resultados para a base EILAT e RSMAS. Observa-se que diferente da base MLC (2008), os modelos baseados em CNNs já superaram os resultados encontrados por Shihavuddin et al. (2013).

Tabela 3.7: Comparação dos resultados de acurácia para as bases: EILAT e RSMAS

	EILAT	**RSMAS**
Gómez-Ríos et al. (2019)	0,980	0,986
Lumini et al. (2019)	0,989	0,992
Shihavuddin et al. (2013)	0,969	0,965

King et al. (2018) utilizaram uma base de dados não pública retirada em uma baía dos Estados Unidos no estado da Flórida. A base possuía 9511 pixels anotados retirados de 1807 imagens e 10 classes: *Acropora palmata*, *Orbicella spp.*, *Siderastrea siderea*, *Porites astreoides*, *Gorgonia ventalina*, *sea plumes*, *sea rods*, *algae*, *rubble* e *sand*.

Para classificação comparou-se os resultados das CNNs: VGG (Simonyan e Zisserman, 2014), InceptionResNetV2 (Szegedy et al., 2016), InceptionV3 (Szegedy et al., 2015) e ResNet (He et al., 2015). Também utilizou-se uma implementação do modelo proposto por Beijbom et al. (2012). A Tabela 3.8 apresenta os resultados encontrados pelos autores.

Tabela 3.8: Resultados de acurácia encontrados por King et al. (2018)

	Acurácia
Resnet152	0,900
Resnet50	0,881
VGG16	0,873
SVM (Beijbom et al., 2012)	0,848
InceptionResNetV2	0,848
InceptionV3	0,847

3.3 SEGMENTAÇÃO SEMÂNTICA DE RECIFES DE CORAIS UTILIZANDO APRENDIZAGEM DE MÁQUINA

Com relação a segmentação semântica, Alonso et al. (2017) utilizaram a base de dados dos autores Beijbom et al. (2016) (Figura 3.3). A base de dados possui as imagens em *Red Green Blue* (RGB) e fluorescente. O *Ground Truth* (GT) é constituído rótulos binários esparsos que avaliam se o píxel é ou não parte de um coral.

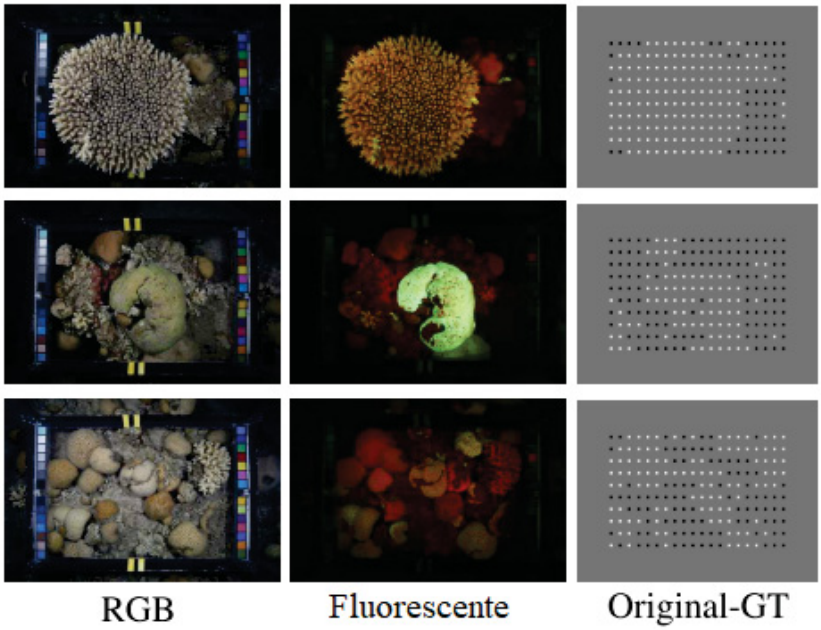


Figura 3.3: Exemplos da base de dados utilizada por Alonso et al. (2017). A base de dados é constituído de imagens RGB, fluorescentes com seus respectivos GT. Fonte: Adaptado de Alonso et al. (2017)

O trabalhou utilizou as técnicas *Simple Linear Iterative Clustering* (SLIC) (Achanta et al., 2012) e *Superpixels Extracted via Energy-Driven Sampling* (SEEDS) (den Bergh et al., 2013) para criar o mapa de segmentação. A Figura 3.4 apresenta um exemplo da criação do mapa de segmentação a partir dos pixels anotados utilizando os algoritmos SLIC e SEEDS. O modelo de aprendizagem de máquina utilizado foi a SegNet (Badrinarayanan et al., 2017), e obteve 0,89 de acurácia por píxel utilizando o SLIC para criação do mapa de segmentação.

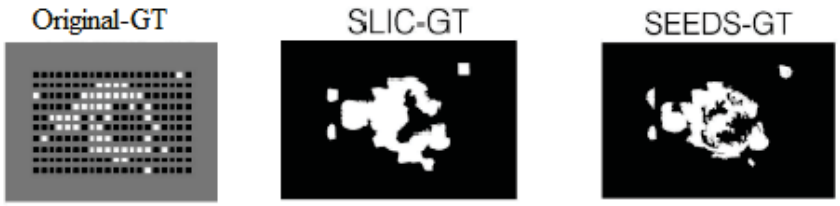


Figura 3.4: SLIC-GT e SEEDS-GT. Fonte: Adaptado de Alonso et al. (2017)

King et al. (2018) em seu trabalho também apresentaram resultados para segmentação semântica utilizando sua base de dados não pública. Utilizou-se o algoritmo SLIC para gerar os mapas de segmentação a partir dos pixels anotados. A base de dados utilizada para segmentação semântica possuía 413 imagens segmentadas semanticamente com as 10 classes utilizadas na classificação.

Os modelos utilizados para realizar a segmentação foram a FCN8s (Long et al., 2014), Dilation8 (Yu e Koltun, 2016), DeepLabV2 (Chen et al., 2016) e uma modificação da Dilation8 proposta pelos autores denominada DilationMod. A Figura 3.5 apresenta uma comparação entre os resultados encontrados pelos modelos. O melhor desempenho foi alcançado pela DeepLabV2 com 67,70 de acurácia por píxel.

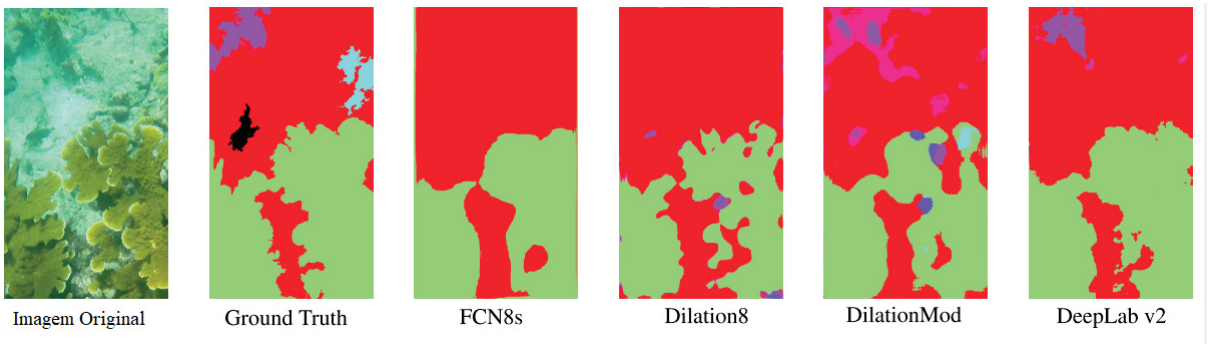


Figura 3.5: Exemplo de uma predição dos modelos testados por King et al. (2018). Fonte: Adaptado de King et al. (2018)

3.4 ANÁLISE CRÍTICA DOS TRABALHOS CORRELATOS

Em relação às bases de dados MLC e PLC um fator que influencia a qualidade das sub-imagens são as anotações serem realizadas por píxel, informando somente que existe o coral naquela região. Então é necessário definir uma resolução para a extração das sub-imagens que provavelmente não será o ideal para todos os casos, pois os corais possuem tamanhos variáveis e não necessariamente essas anotações se encontram no centro do coral.

Em relação à classificação de corais observa-se uma tendência de se utilizar CNNs. As bases EILAT e RSMAS as CNNs pré-treinadas possuem desempenho melhor que as técnicas tradicionais. Entretanto, na base MLC (2008) os resultados das CNNs somente se aproximam dos encontrados por Shihavuddin et al. (2013).

Na literatura não existem modelos que utilizaram *transfer learning* entre as bases MLC, EILAT, RSMAS somente a inicialização com os valores pré-treinados na ImageNet. Analogamente, não existem na literatura trabalhos que utilizaram imagens nas quais os corais não são necessariamente a informação central da imagem. Todos os trabalhos realizam algum tratamento para reduzir a quantidade de informações extras presentes nas imagens.

Em relação à parte de segmentação semântica, os trabalhos realizados por King et al. (2018) e Alonso et al. (2017) não utilizaram um mapa de segmentação real que compromete a análise dos resultados e não utilizaram a métrica mIoU (Equação 2.25) para avaliar os modelos. Portanto, os resultados apresentados nos trabalhos podem estar superestimados devido a utilização da métrica acurácia por píxel. Contudo, são dois resultados que demonstram que existe espaço para evolução e melhora no desenvolvimento de modelos para realizar a segmentação semântica de imagens de corais.

Por fim, muitos trabalhos citados não possuem uma metodologia clara entre treino, validação e teste, o que impossibilita a reprodutibilidade dos trabalhos mesmo que a base de dados seja pública. Metodologias claras em relação a esses pontos incentivam os testes de novos modelos e novas técnicas para melhorar os resultados encontrados anteriormente.

4 METODOLOGIA EXPERIMENTAL

A metodologia utilizada no atual trabalho se dividiu nas seguintes etapas:

- Coleta, limpeza, padronização e separação da base de dados (Seção 4.1)
- Tarefa de classificação (Seção 4.2)
- Tarefa de segmentação e de localização (Seção 4.3)

Para realizar os testes, o atual trabalho utilizou duas máquinas presentes no Departamento de Informática (DINF) da Universidade Federal do Paraná (UFPR). A Tabela 4.1 apresenta as especificações das duas máquinas utilizadas para executar os experimentos.

Tabela 4.1: Especificações (sistema operacional, memória RAM, CPU, GPU) das máquinas utilizadas para a realização dos experimentos

	Pilheira	**Tesla2**
Sistema Operacional	Ubuntu 18.04.5 LTS	Ubuntu 18.04.6 LTS
Memória RAM	126 GB	133 GB
CPU	Intel(R) Xeon(R) Silver 4114 2.20GHz	Quad-Core AMD Opteron(tm) 8387
GPU	2x TITAN Black 6GB	TITAN X 12GB + TITAN Xp 12GB

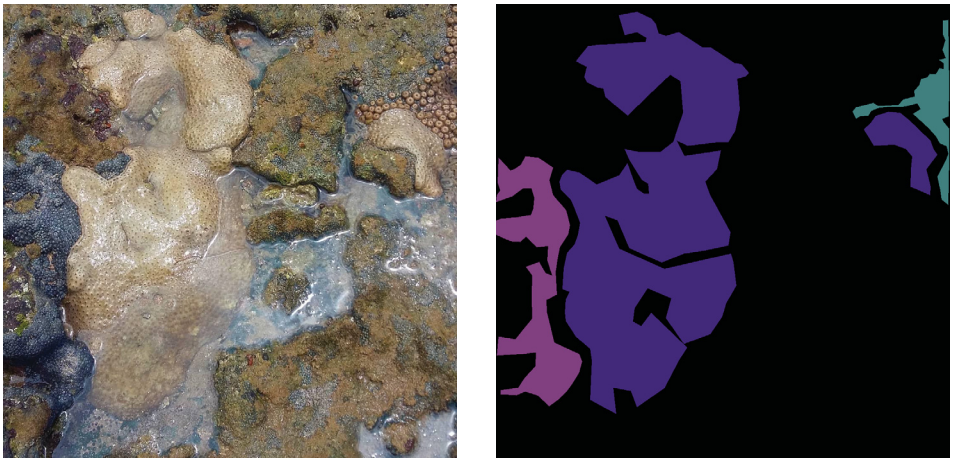
Para o desenvolvimento dos experimentos utilizou-se as bibliotecas *TensorFlow* 2.4.1 (Abadi et al., 2015) e *Sklearn* 0.24.2 (Buitinck et al., 2013) implementadas em *Python*. Selecionou-se a versão 3.8.8 do *Python*.

4.1 BASE DE DADOS

A base de dados foi construída em parceria com os pesquisadores do LECOM da UFRN. As imagens foram extraídas utilizando a ferramenta *Instaloader* (Graf et al., 2021) que realiza o download das fotos do Instagram com a *hashtag* #DeOlhoNosCorais. A rotulação foi realizada pelos pesquisadores do LECOM utilizando a ferramenta *Labelme* (Wada, 2016). Além das imagens retiradas do Instagram a base conta com algumas imagens extras cedidas pelos pesquisadores do LECOM. Essas imagens foram realizadas por mergulhadores do grupo de pesquisa.

As imagens foram organizadas removendo-se imagens duplicadas utilizando a ferramenta *FiftyOne* (Moore e Corso, 2020), padronizando-se o nome das classes e verificando-se os mapas de segmentações corrompidos. Atualmente a base conta com 1411 imagens com 21 classes com imagens retiradas datadas de 2014 a 2021. Comparando com as bases da literatura apresentadas na Tabela 3.2, a atual base de dados possuí mais imagens que as bases EILAT e RSMAS. Entretanto, o número de classes das bases EILAT e RSMAS é inferior com 8 e 14 classes, respectivamente.

O diferencial da base de dados em relação as outras bases presentes na literatura é presença dos mapas de segmentação (Figura 4.1(b)) para cada imagem. Rotulou-se também as imagens com as suas respectivas classes principais (Figura 4.1(a)). Diferentemente de outras bases citadas na literatura como MLC, PLC, RSMAS e EILAT, a base deste trabalho é composta



(a) (b)

Figura 4.1: Rotulação da base de dados - (a) imagem original - classe principal: *Palythoa caribaeorum* (b) mapa de segmentação - rosa: *Zoanthus sociatus*, roxo: *Palythoa caribaeorum*, azul: *Palythoa spp*

somente de corais já que não há a rotulação para organismos como algas, esponjas ou minerais como areia que estão presentes no ecossistema dos recifes.

Devido à presença do mapa de segmentação e o rótulo da classe principal, a base de dados pode ser utilizada para diferentes tarefas de visão computacional (Figura 2.11). Logo, a base foi organizada para as seguintes tarefas:

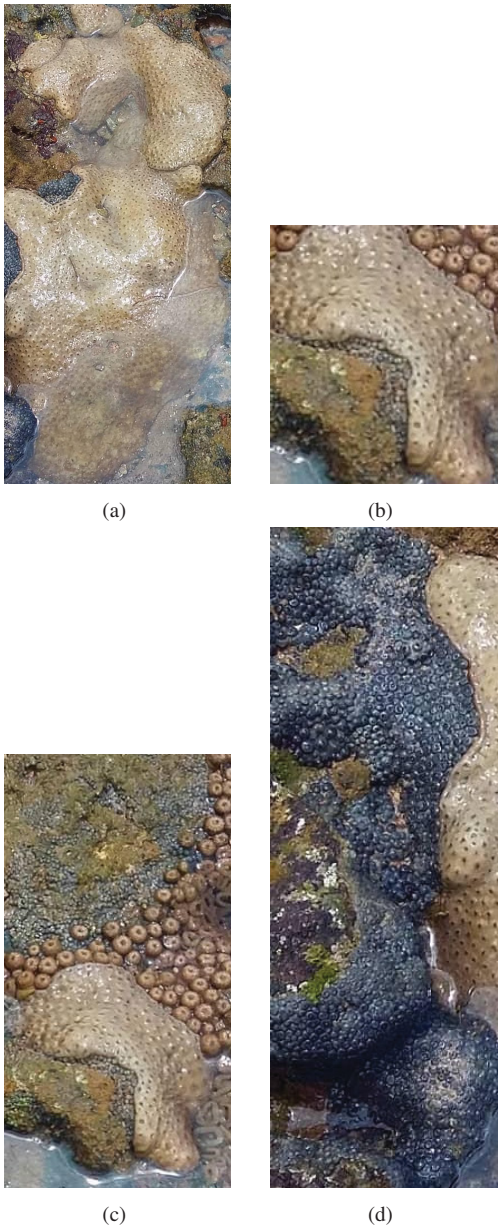
- Classificação
- Segmentação semântica
- Segmentação instanciada
- Localização de objetos

Criou-se também uma nova base para a tarefa de classificação a partir das *bounding boxes* presentes na base de localização de objetos para retirar sub-imagens da imagem inteira que contém somente o coral. A Figura 4.2 apresenta um exemplo dessa extração utilizando a Figura 4.1(a). Essa nova base possui menos ruídos que a base composta pelas imagens inteiras e apresenta um número maior de imagens, pois uma mesma imagem original pode conter um ou mais corais extraídos nas sub-imagens.

A base final conta com imagens retiradas do Instagram com a *hashtag* "#DeOlhoNosCorais" e algumas imagens extras, realizadas por mergulhadores do LECOM. Essas imagens extras foram utilizadas somente para aumentar a base de treinamento. Como o trabalho avaliou a viabilidade de se utilizar modelos de aprendizagem de máquina para rotular automaticamente as imagens extraídas do Instagram, a divisão das bases de treinamento, validação e teste seguiu os seguintes critérios:

- Base de treinamento: imagens datadas até 31/12/2018 + imagens extra do LECOM
- Base de validação: imagens datadas de 01/01/2019 até 30/06/2019
- Base de teste: imagens datadas de 01/07/2019 até 24/08/2021

A escolha de se utilizar a data como critério é avaliar como o modelo iria performar em uma situação real de monitoramento. A base de teste contempla um intervalo de 2 anos, devido à queda do número de postagens durante a pandemia de Covid-19.



(a) (b)

(c) (d)

Figura 4.2: Sub-Imagens da Figura 4.1(a). rótulos: (a) *Palythoa caribaeorum* (b) *Palythoa caribaeorum* (c) *Palythoa spp* (d) *Zoanthus sociatus*

4.1.1 Distribuição da Base de dados

Para criação da sub-base para a tarefa de classificação utilizaram-se os rótulos apresentados na Figura 4.1(a) que possuí a classe principal da imagem. A Tabela 4.2 apresenta a distribuição da base de classificação. A Figura 4.3 apresenta um gráfico de barra da distribuição da base de dados ordenado pelo número total de imagens.

Para criação das sub-bases para as tarefas de segmentação semântica utilizaram-se os mapas de segmentação apresentados na Figura 4.1(b) e o *Labelme*. Para a base de segmentação semântica utilizou o formato do PASCAL VOC (Everingham et al., 2012) e para a base de segmentação instanciada adotou-se o formato da base COCO (Lin et al., 2014).

A Tabela 4.3 apresenta a distribuição da base para as tarefas de segmentação semântica e segmentação instanciada.

Tabela 4.2: Distribuição da base de dados utilizando a classe principal das imagens

Classes	**Treino**	**Validação**	**Teste**
1- *Agaricia spp*	64	5	5
2- *Favia gravida*	57	6	1
3- *Madracis decactis*	6	1	3
4- *Meandrina braziliensis*	0	0	4
5- *Millepora alcicornis*	114	20	33
6- *Millepora braziliensis*	0	1	4
7- *Montastraea cavernosa*	250	29	34
8- *Mussismilia braziliensis*	7	8	6
9- *Mussismilia harttii*	13	8	19
10- *Mussismilia hispida*	36	39	78
11- *Mussismilia leptophylla*	0	1	1
12- *Palythoa caribaeorum*	85	21	59
13- *Palythoa spp*	0	3	8
14- *Parazoanthus swiftii*	2	0	0
15- *Porites astreoides*	50	8	9
16- *Porites branneri*	2	1	2
17- *Scolymia wellsi*	1	7	11
18- *Siderastrea stellata*	168	17	27
19- *Tubastraea spp*	20	4	22
20- *Zoanthus sociatus*	10	1	9
21- *Zoanthus spp*	4	1	6
Total	**889**	**181**	**341**

Tabela 4.3: Distribuição da base de dados utilizando o mapa de segmentação

	Treino	**Validação**	**Teste**
Imagens	889	181	341

Para as duas tarefas foram criadas duas bases: com os rótulos binários e com os rótulos categóricos. Como mostrado na Figura 4.4, nos mapas categóricos existe a diferenciação entre as espécies de corais e na binária somente existem as classes: Coral e Plano de Fundo.

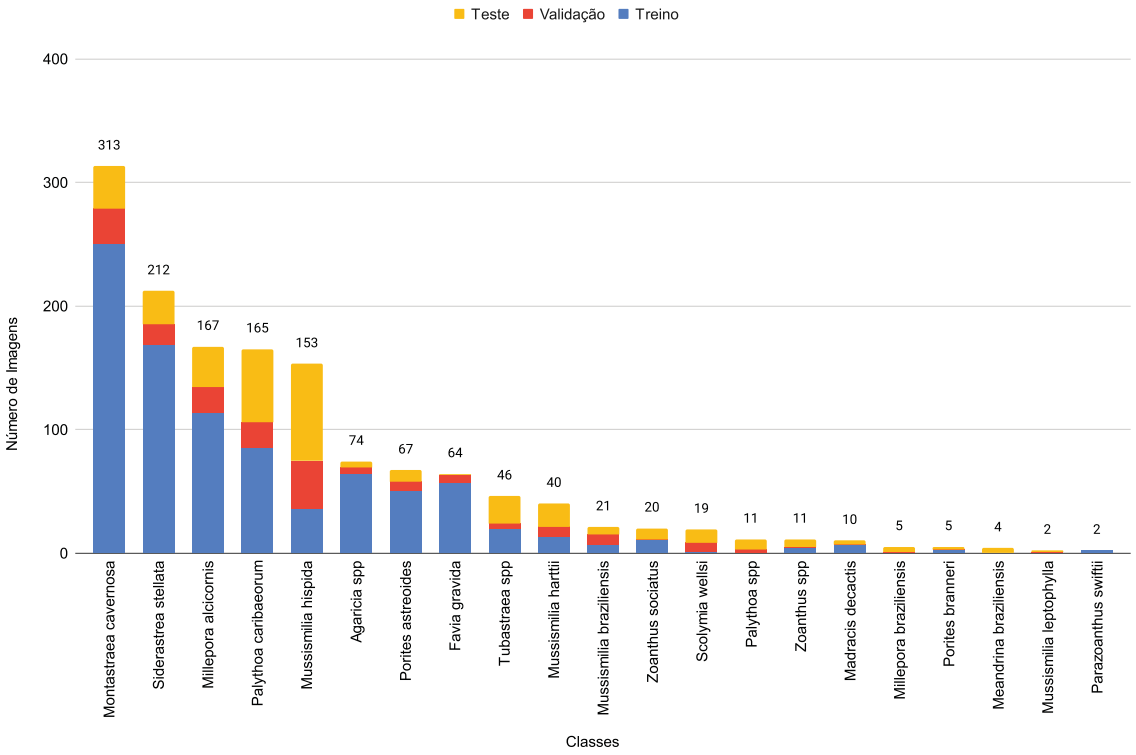


Figura 4.3: Distribuição da base de dados utilizando a classe principal das imagens ordenada pelo número de imagens totais

(a) (b) (c)

Figura 4.4: Modificação da Figura 4.1(a) para a tarefa de segmentação semântica - (a) imagem original (b) mapa de segmentação categórico - rosa: *Zoanthus sociatus*, roxo: *Palythoa caribaeorum*, Azul: *Palythoa spp* (c) mapa de segmentação binário - vermelho: coral

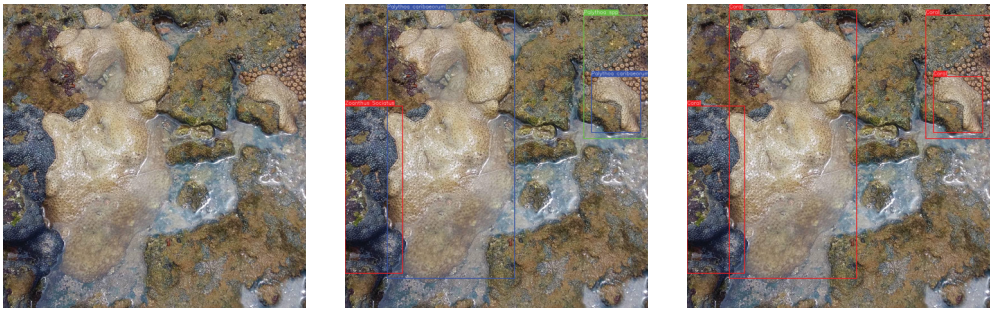
Para as bases de segmentação instanciada, segue os mesmos modelos dos utilizados na segmentação semântica como mostrado na Figura 4.5.

Para a sub-base para a tarefa de localização de objetos utilizou-se a ferramenta *FiftyOne* para converter a base de segmentação instanciada em uma base de localização de objetos no formato da Yolov5 (Jocher, 2020). A distribuição da base permaneceu a mesma mostrada na Tabela 4.3. A base de localização de objetos também possui os rótulos categóricos e binário, conforme apresentados na Figura 4.6.

A distribuição da base utilizando sub-imagens está apresentado na Tabela 4.4. Para essa base o número de classe sobe para 22 classes, com adição da classe *Palythoa grandiflora* que não possui nenhuma imagem inteira rotulada com essa classe.

(a) (b) (c)

Figura 4.5: Modificação da Figura 4.1(a) para a tarefa de segmentação instanciada - (a) imagem original (b) segmentação instanciada categórica (c) segmentação instanciada binária



(a) (b) (c)

Figura 4.6: Modificação da Figura 4.1(a) para a tarefa de localização de objetos - (a) imagem original (b) localização de objetos categórica (c) localização de objetos binária

A Figura 4.7 apresenta um gráfico de barra da distribuição da base de dados ordenado pelo número total de imagens. Observa-se que existem classes predominantes na base e existe uma diferença de distribuição entre as bases de treino, teste e validação.

4.2 METODOLOGIA PARA A REALIZAÇÃO DA TAREFA DE CLASSIFICAÇÃO

Utilizando as bases construídas para classificação com as imagens inteiras e as sub-imagens avaliou-se duas metodologias utilizando CNNs:

- Extração de características utilizando CNNs + classificador
- CNN *end-to-end* (treinamento completo)

A Figura 4.8 apresenta a diferença entre as duas metodologias para uma predição utilizando uma CNN genérica. Na extração de característica + classificador (Figura 4.8(b)) a CNN comporta-se como um extrator de característica. Nesta configuração a CNN é utilizada sem a sua última camada densa, que realiza a predição, e utilizam-se seus pesos pré-treinados em outra base, como a ImageNet (Deng et al., 2009). O classificador é treinado com o vetor de características proveniente da CNN. Na metodologia CNN *end-to-end* (Figura 4.8(a)), a CNN é treinada na base e realiza todo o processo de extração de características e classificação.

A metodologia de testes foram realizadas utilizando duas etapas:

1. Ajuste de parâmetros e seleção dos modelos utilizando as bases de treinamento e validação

Tabela 4.4: Distribuição da base de dados utilizando as sub-imagens

Classes	Treino	Validação	Teste
1- *Agaricia spp*	101	14	9
2- *Favia gravida*	108	6	4
3- *Madracis decactis*	28	1	26
4- *Meandrina braziliensis*	0	0	21
5- *Millepora alcicornis*	170	54	93
6- *Millepora braziliensis*	0	3	0
7- *Montastraea cavernosa*	399	57	95
8- *Mussismilia braziliensis*	18	45	20
9- *Mussismilia harttii*	61	57	67
10- *Mussismilia hispida*	78	57	149
11- *Mussismilia leptophylla*	0	1	3
12- *Palythoa caribaeorum*	288	67	194
13- *Palythoa grandifllora*	2	0	0
14- *Palythoa spp*	4	11	26
15- *Parazoanthus swiftii*	6	0	0
16- *Porites astreoides*	79	21	25
17- *Porites branneri*	2	1	2
18- *Scolymia wellsi*	2	8	13
19- *Siderastrea stellata*	741	60	113
20- *Tubastraea spp*	119	25	256
21- *Zoanthus sociatus*	90	4	26
22- *Zoanthus spp*	2	3	3
Total	**2298**	**495**	**1145**

2. Avaliação dos classificadores utilizando a base de teste.

Na primeira etapa utilizou-se somente a base de treinamento e validação para ajustar os parâmetros e selecionar os melhores modelos. Por fim, na etapa de Teste utilizou-se a base de treinamento e validação como treinamento e os parâmetros definidos na primeira etapa para avaliar esses modelos frente a base de teste.

Para a seleção dos modelos, além da avaliação das duas metodologias apresentadas na Figura 4.8, foi realizado uma combinação entre os modelos selecionados que utiliza a média das predições de cada modelo treinado para realizar uma nova inferência. A combinação entre modelos visa melhor o desempenho da classificação. Os classificadores possuem desempenhos diferentes para cada classe do problema, então a junção das suas predições pode gerar um classificador que se beneficia dos melhores resultados de cada classificador.

4.2.1 Treinamento CNN

Para a inicialização das CNNs com os pesos pré-treinados na ImageNet utilizou-se os valores disponíveis no *TensorFlow*. O treinamento das CNNs foram divididos em duas partes:

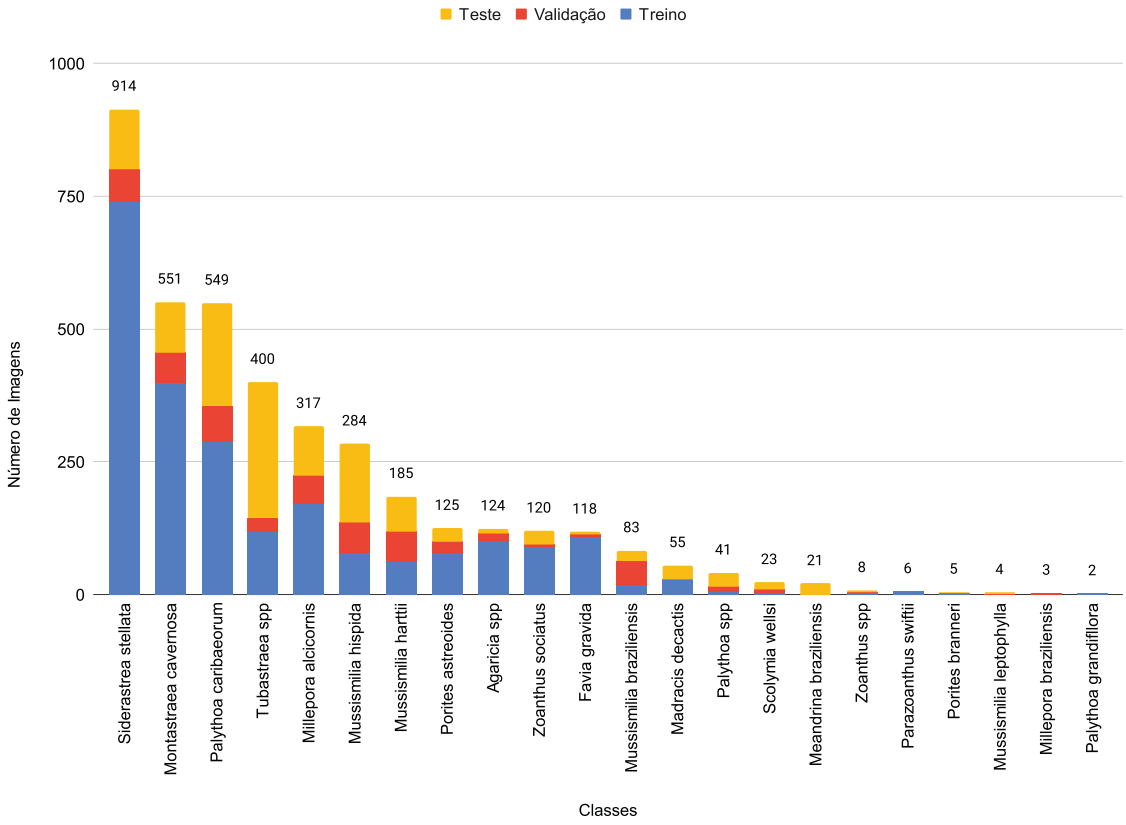


Figura 4.7: Distribuição da base de dados utilizando as sub-imagens ordenadas pelo número de imagens totais

1. Inicialização da camada densa
2. Treinamento da CNN completa

Ao realizar a inicialização da camada densa antes de se treinar a CNN por completo visa evitar a degradação das camadas convulsionais pré-treinadas na ImageNet. Uma diferença entre às duas etapas é que na primeira etapa não se aplica o decaimento exponencial nas primeiras 20 épocas de treinamento.

A Figura 4.9 apresenta um gráfico de perda (Equação 2.9) em relação ao número de épocas utilizando às duas etapas (inicialização da camada densa e treinamento da CNN completa) em conjunto. No treinamento utiliza-se o *early stopping* para encerrar o treinamento antes do modelo entrar em *overfitting* em ambas as etapas. Para a regularização do modelo aplicou-se *Dropout* na camada densa.

No ajuste de parâmetros utilizando a base de validação para as CNNs procurou-se encontrar os seguintes parâmetros para às duas etapas de treinamento:

- Taxa de aprendizado
- Decaimento
- *Momentum*
- Número de épocas de treinamento

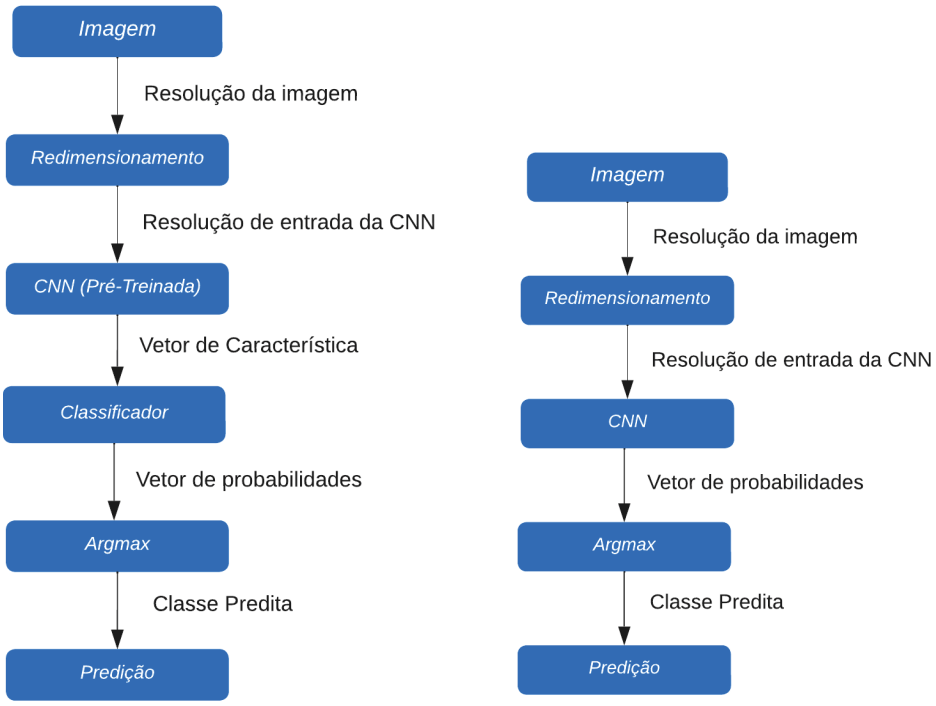


Figura 4.8: Comparação das duas metodologias utilizadas na tarefa de classificação - (a) extração de características (CNNs) + classificador (b) CNNs

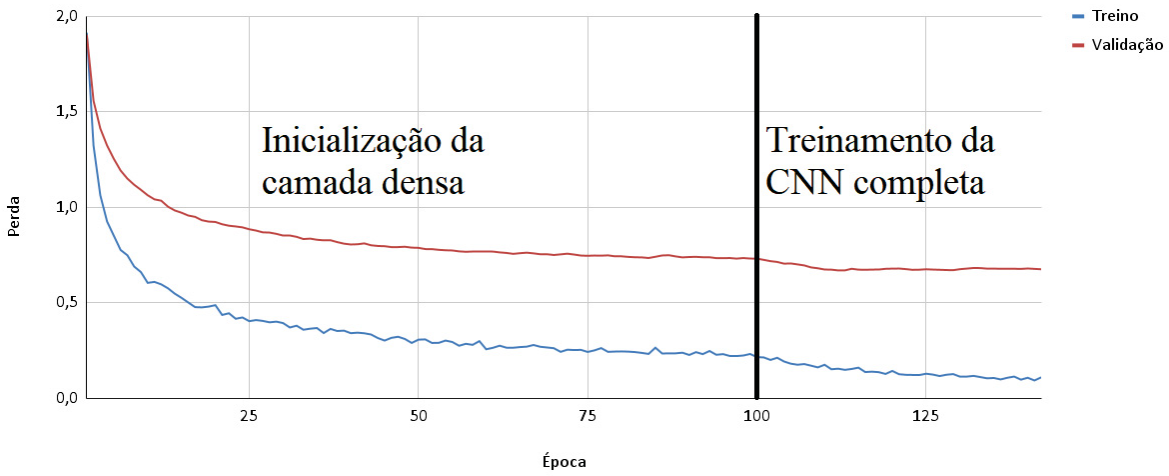


Figura 4.9: Exemplo do gráfico de perda do modelo para as bases de treinamento e teste em função do número de épocas de treinamento

- *Batch size*
- *Dropout*

Para a seleção dos modelos utilizou-se o valor da função de custo (Equação 2.9), pois ele avalia o nível de confiança do modelo considerando a predição de todas as classes do problema. As demais métricas foram utilizas para as análises finais dos resultados na base de teste.

Aplicou-se *data augmentation* na base de treinamento para se gerar novas imagens durante as épocas de treinamento do modelo. Selecionou-se a operação de giro (Figura 4.10). A

operação de giro é popularmente utilizada em treinamento de CNNs. Ela inverte a imagem na horizontal e na vertical de forma aleatória.

(a) (b) (c)

Figura 4.10: Exemplo da operação de giro na vertical e na horizontal. (a) Imagem original, (b) operação de giro na horizontal e (c) operação de giro na vertical

Em relação à inicialização das camadas convulsionais avaliou-se duas metodologias de *transfer learning*:

1. ImageNet
2. PLC (ImageNet + PLC)

Na metodologia 1 utilizaram-se somente os modelos pré-treinados na ImageNet para o treinamento na base de dados do atual trabalho. Em contraste, na metodologia 2 utilizou a base PLC como intermediário. Sendo assim, treinaram-se os modelos na base PLC com os pesos pré-treinados na ImageNet para serem posteriormente treinados na base de dados do atual trabalho.

A base PLC foi escolhida devido ao seu número de imagens em comparação com as bases EILAT e RSMAS e uma maior diversidade de regiões em comparação a MLC. Portanto, o objetivo foi que a CNN treinada utilizando a PLC como intermediária possua camadas, como apresentado na Figura 2.9, que reconheçam melhor os padrões presentes no ecossistema dos recifes de corais.

A distribuição da base PLC e os resultados estão presentes no anexo A. Selecionou-se 12 classes das 20 presentes na PLC e retirou-se sub-imagens de 224 × 224 centradas no píxel rotulado.

4.2.2 Experimento com imagens inteiras

Para as imagens inteiras, primeiramente filtrou-se a base de dados para as classes que possuíam pelo menos 50 imagens de treinamento, a Tabela 4.5 apresenta a distribuição e as classes utilizadas nessa etapa.

Os modelos utilizados nesta etapa foram:

- EfficientNetB7 (Tan e Le, 2019) + regressão logística
- ResNet101 (ImageNet)
- ResNet101 (PLC)

Os parâmetros utilizados para o treinamento das ResNet101 estão apresentados na Tabela 4.6. A resolução de entrada para às duas configurações foi de 224 × 224. Para a EfficientNetB7, utilizou-se a resolução de entrada de 600 × 600 que retorna um vetor de característica para a

Tabela 4.5: Distribuição da base de dados para a classificação das imagens inteiras

Classes	**Treino**	**Validação**	**Teste**
1- *Agaricia spp*	64	5	5
2- *Favia gravida*	57	6	1
3- *Millepora alcicornis*	114	20	33
4- *Montastraea cavernosa*	250	29	34
5- *Palythoa caribaeorum*	85	21	59
6- *Porites astreoides*	50	8	9
7- *Siderastrea stellata*	168	17	27
Total	**788**	**106**	**168**

Tabela 4.6: Parâmetros de treinamento das ResNet101 para a classificação das imagens inteiras

		SGD			**Modelo**		
Modelo	Etapas de treinamento	Taxa de aprendizado	*Momentum*	Decaimento	Batch size	Épocas	Dropout
ResNet101 (ImageNet)	Camada Densa	10^{-4}	0,9	0,005	32	100	0,2
	CNN Completa	10^{-5}	0,9	0,020	32	32	0,2
ResNet101 (PLC)	Camada Densa	$5x10^{-5}$	0,9	0,005	32	100	0,2
	CNN Completa	$5x10^{-6}$	0,9	0,020	32	48	0,2

regressão logística ($C = 1$) de tamanho 2560. Essas resoluções são as mesmas que os modelos foram treinados na ImageNet.

Por fim, os resultados foram analisados utilizando o LIME (Seção 2.8.2) para avaliar quais são as regiões que mais influenciam na predição do modelo. O algoritmo auxiliou na interpretação dos resultados encontrados mostrando pontos bons e ruins da predição do modelo e, posteriormente, avaliar-se possíveis pontos de melhoria em projetos futuros. A utilização do LIME também auxilia na interpretação dos resultados para pessoas que não possuem um conhecimento técnico sobre a área de aprendizagem de máquina.

4.2.3 Experimento com sub-imagens

Para as sub-imagens foram realizados dois filtros na base apresentada na Tabela 4.4. O primeiro filtro foi o número de pixels das imagens, definiu-se dois limiares: 50176 (224 × 224) pixels e 16384 (128 × 128) pixels. Esses limiares foram selecionados, devido à resolução de entrada das CNNs. Posteriormente selecionou-se as classes com pelo menos 50 imagens na base de treinamento. As Tabelas 4.7 e 4.8 apresentam a distribuição das duas bases geradas.

Os modelos utilizados para a base com limiar de (224 × 224) foram:

- EfficientNetB0 (Tan e Le, 2019) + regressão logística
- ResNet101 (ImageNet)
- ResNet101 (PLC)

Os parâmetros utilizados para o treinamento dos modelos estão apresentados na Tabela 4.9. A regressão logística ($C = 1$) foi treinada com um vetor de característica de tamanho 1280 da EfficientNetB0. Foi selecionada a resolução de entrada de 224 x 224 para todas as CNNs.

Tabela 4.7: Distribuição da base de dados para a classificação das sub-imagens (224 × 224)

Classes	**Treino**	**Validação**	**Teste**
1- *Agaricia spp*	85	12	9
2- *Favia gravida*	79	6	4
3- *Millepora alcicornis*	137	27	40
4- *Montastraea cavernosa*	303	31	51
5- *Palythoa caribaeorum*	249	37	92
6- *Porites astreoides*	58	16	12
7- *Siderastrea stellata*	432	11	24
8 - *Zoanthus sociatus*	51	3	15
Total	**1394**	**143**	**247**

Tabela 4.8: Distribuição da base de dados para a classificação das sub-imagens (128 × 128)

Classes	**Treino**	**Validação**	**Teste**
1- *Agaricia spp*	91	13	9
2- *Favia gravida*	89	6	4
3- *Millepora alcicornis*	155	35	53
4- *Montastraea cavernosa*	341	36	65
5- *Mussismilia hispida*	51	45	103
6- *Palythoa caribaeorum*	269	50	135
7- *Porites astreoides*	60	20	19
8- *Siderastrea stellata*	566	27	48
9- *Zoanthus sociatus*	71	3	20
Total	**1693**	**235**	**456**

Tabela 4.9: Parâmetros de treinamento das ResNet101 para a classificação das sub-imagens (224 x 224)

		SGD			**Modelo**		
Modelo	Etapas de treinamento	Taxa de aprendizado	*Momentum*	Decaimento	Batch size	Épocas	Dropout
ResNet101 (ImageNet)	Camada Densa	10^{-4}	0,9	0,005	32	100	0,2
	CNN Completa	10^{-5}	0,9	0,020	32	12	0,2
ResNet101 (PLC)	Camada Densa	$5x10^{-5}$	0,9	0,005	32	100	0,2
	CNN Completa	$5x10^{-6}$	0,9	0,020	32	40	0,2

Os modelos utilizados para a base com limiar de (128 × 128) foram:

- MobileNetV2 (Sandler et al., 2018) + regressão logística
- MobileNetV2 (ImageNet)

Os parâmetros utilizados para o treinamento da MobileNetV2 (ImageNet) estão apresentados na Tabela 4.10. A MobileNetV2 foi utilizada com resolução de entrada igual a 128 × 128

nas duas metodologias. A regressão logística ($C = 1$) foi treinada com um vetor de característica de tamanho 1280 proveniente da MobileNetV2.

Tabela 4.10: Parâmetros de treinamento da MobileNetV2 para as sub-imagens 128 x 128

	SGD			**Modelo**		
Etapas de treinamento	Taxa de Aprendizado	*Momentum*	Decaimento	Batch size	Épocas	Dropout
Camada Densa	10^{-4}	0,9	0,005	32	100	0,2
CNN Completa	10^{-5}	0,9	0,020	32	119	0,2

4.3 TAREFA DE SEGMENTAÇÃO E DE LOCALIZAÇÃO

Realizou-se também experimentos para as tarefas de segmentação semântica e localização de objetos com os rótulos binários. Foram utilizadas todas as imagens na distribuição apresentada na Tabela 4.3.

Para a tarefa segmentação semântica utilizou-se uma modificação da U-Net denominada U-net (Pix2Pix) (TensorFlow, 2020), que tem como *enconder* a MobileNetV2 (Sandler et al., 2018) e *decoder* a Pix2Pix (Isola et al., 2016). Para a tarefa de localização, utilizou-se a Yolov5 (Jocher, 2020).

Os parâmetros definidos para a U-net (Pix2Pix) e para a Yolov5 estão apresentados na Tabela 4.11 e 4.12, respectivamente. Para a Yolov5, os parâmetros não listados tiveram os seus valores padrões mantidos.

Tabela 4.11: Parâmetros de treinamento da U-Net (Pix2Pix) para a tarefa de segmentação semântica binária

Adam	**Modelo**	
Taxa de Aprendizado	Batch size	Épocas
10^{-3}	32	20

Tabela 4.12: Parâmetros de treinamento da Yolov5 para a tarefa de localização de objetos

Tamanho da Imagem	Batch size	Épocas
640	8	85

5 RESULTADOS EXPERIMENTAIS

Este capítulo apresenta os resultados e as discussões para as tarefas de classificação (Seção 5.1), segmentação semântica e localização de objetos (Seção 5.2).

5.1 AVALIAÇÃO DOS CLASSIFICADORES

5.1.1 Experimento com imagens inteiras

A Tabela 5.1 apresenta os resultados dos modelos para a base contendo as imagens inteiras.

Tabela 5.1: Resultados encontrados no experimento utilizando as imagens inteiras para as bases de validação e teste

	Acurácia		**F1**		**MCC**	
	Val.	Teste	Val.	Teste	Val.	Teste
EfficientNetB7 + regressão logística	0,82	**0,71**	**0,80**	0,69	0,78	**0,63**
ResNet101 (ImageNet)	0,73	0,59	0,63	0,53	0,67	0,48
ResNet101 (PLC)	0,67	0,58	0,62	0,51	0,60	0,47
Combinação	**0,83**	0,67	0,75	**0,72**	**0,79**	0,58

Observa-se que existe uma queda de desempenho entre a base de validação e a base de teste, mostrando que o número de imagens ainda não é o suficiente para uma correta generalização dos modelos. O modelo (EfficientNetB7 + regressão logística) se destaca nos resultados dentre as métricas selecionadas, no entanto a combinação entre os 3 modelos selecionados obteve sucesso em melhorar alguns resultados tanto na base de validação quanto na base de teste.

A Figura 5.1 apresenta as matrizes de confusão do modelo (EfficientNetB7 + regressão logística) para a base de teste. Observa-se um baixo desempenho na base para as classes: *Montastraea cavernosa*, *Palythoa caribaeorum* e *Porites astreoides*. Segundo a Tabela 4.5 essas 3 classes representam 60% da base de teste.

Para interpretar os resultados utilizando o LIME, selecionou-se 3 imagens da base de teste (Figura 5.2). A Figura 5.2(a) pertence a uma classe que os modelos obtiveram resultados satisfatórios, a Figura 5.2(b) pertence a uma classe que os modelos tiveram dificuldade de classificar e a Figura 5.2(c) é uma imagem que contém texto.

A Tabela 5.2 apresenta as probabilidades preditas pelos modelos para a Figura 5.2(a). Observa-se que, para essa imagem, todos os 3 modelos predizem a classe corretamente e possuem alta probabilidade na classe 1.

Tabela 5.2: Probabilidades preditas pelos modelos para a Figura 5.2(a). Classe com maior probabilidade (classe 1), segunda classe com maior probabilidade (classe 2)

	Classe 1		**Classe 2**	
	Classe	Prob.	Classe	Prob.
EfficientNetB7 + regressão logística	*Millepora alcicornis*	0,990	*Palythoa caribaeorum*	0,010
ResNet101 (ImageNet)	*Millepora alcicornis*	0,990	*Porites astreoides*	0,010
ResNet101 (PLC)	*Millepora alcicornis*	0,999	*Favia gravida*	0,001

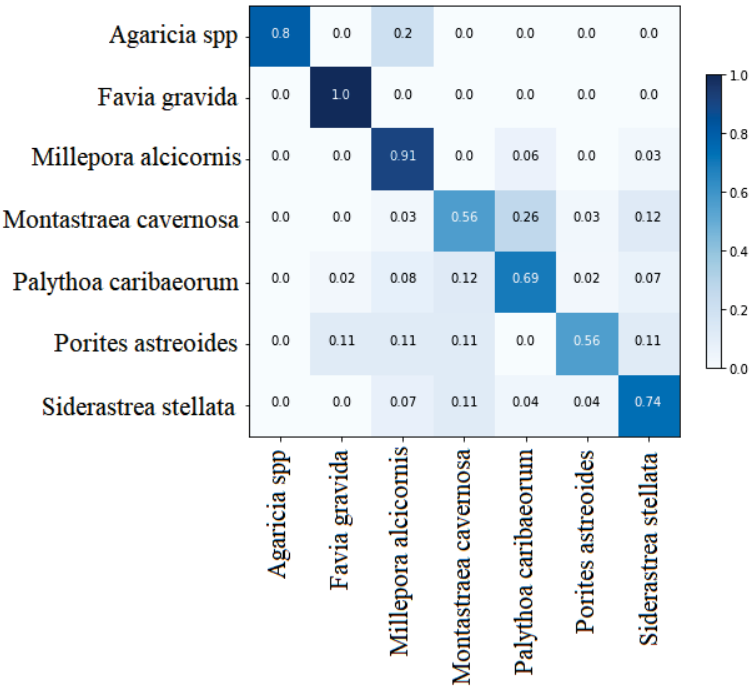


Figura 5.1: Matriz de confusão do modelo EfficientNetB7 + regressão logística para a base de teste das imagens inteiras

(a) (b) (c)

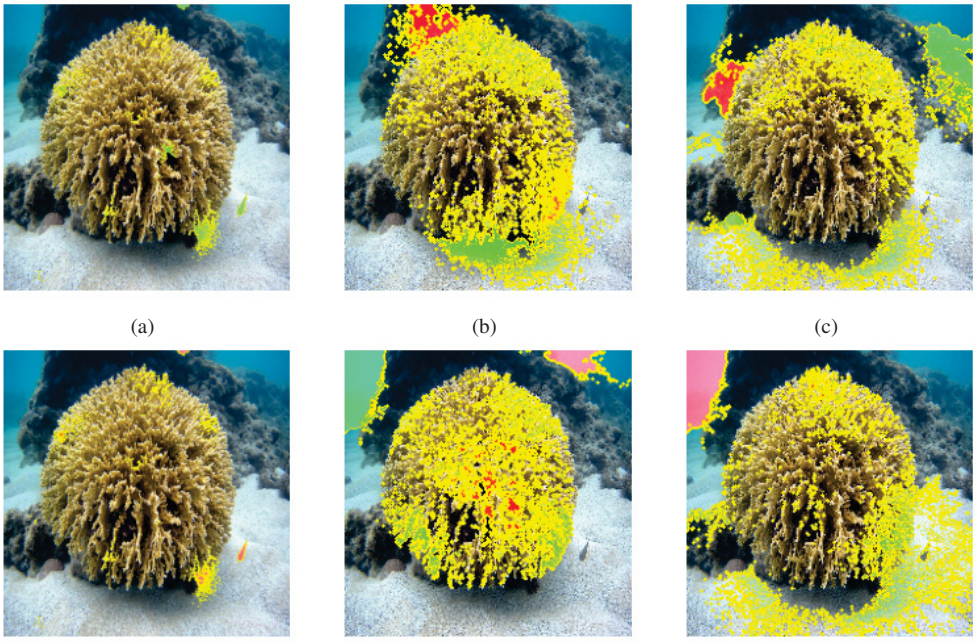
Figura 5.2: Imagens utilizadas em conjunto com o LIME. Classes: (a) *Millepora alcicornis*, (b) *Montastraea cavernosa*, (c) *Palythoa caribaeorum*

A Figura 5.3 apresenta os resultados do LIME para as duas classes prováveis dos 3 modelos. Os modelos em geral possuem um foco no coral centralizado da imagem, entretanto, os modelos baseados na ResNet101 interagem com outras regiões da imagem.

A Tabela 5.3 apresenta as probabilidades preditas pelos modelos para a Figura 5.2(b). Observa-se que, para essa imagem, todos os 3 modelos a predizem incorretamente e eles predizem a mesma classe (*Siderastrea stellata*). Contudo, o modelo (EfficientNetB7 + regressão logística) possui a classe correta (*Montastraea cavernosa*) com a sua Classe 2 com 0,45 de probabilidade.

A Figura 5.4 apresenta os resultados do LIME para as duas classes prováveis dos 3 modelos em relação à Figura 5.2(b). Mesmo que incorretas, as predições feitas pelos modelos baseiam-se no coral da imagem ao passo que também interagem com outras partes da imagem.

A Tabela 5.4 apresenta as probabilidades preditas pelos modelos para a Figura 5.2(c). Observa-se que, para essa imagem todos, os 3 modelos a predizem incorretamente. A EfficientNetB7 + regressão logística prediz a classe *Siderastrea stellata* e as ResNet101 a classe *Porites*



(a) (b) (c)

(d) (e) (f)

Figura 5.3: Resultados do LIME para Figura 5.2(a) utilizando as duas classes mais prováveis dos modelos. Os pontos em verde e vermelho indicam que a região influência positivamente e negativamente, respectivamente, na predição do modelo para a classe avaliada. (a) EfficientNetB7 + regressão logística - Classe 1 (b) ResNet101 (ImageNet) - Classe 1 (c) ResNet (PLC) - Classe 1 (d) EfficientNetB7 + regressão logística - Classe 2 (e) ResNet101 (ImageNet) - Classe 2 (c) ResNet (PLC) - Classe 2

Tabela 5.3: Probabilidades preditas pelos modelos para a Figura 5.2(b). Classe com maior probabilidade (classe 1), segunda classe com maior probabilidade (classe 2)

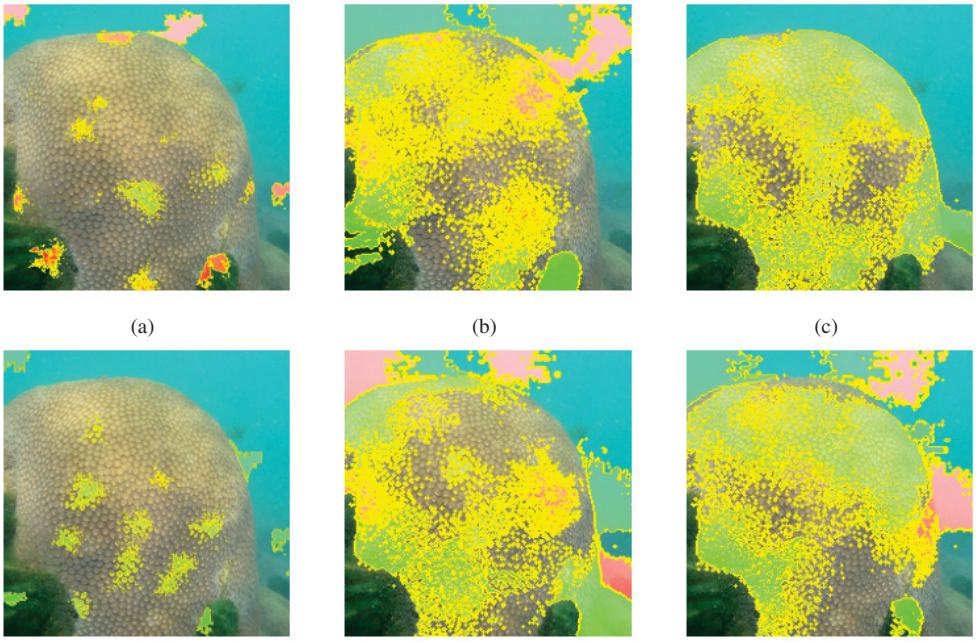
	Classe 1		**Classe 2**	
	Classe	Prob.	Classe	Prob.
EfficientNetB7 + regressão logística	*Siderastrea stellata*	0,540	*Montastraea cavernosa*	0,450
ResNet101 (ImageNet)	*Siderastrea stellata*	0,970	*Palythoa caribaeorum*	0,020
ResNet101 (PLC)	*Siderastrea stellata*	0,999	*Montastraea cavernosa*	0,001

astreoides. Apesar disso, os modelos EfficientNetB7 + regressão logística e ResNet101 (PLC) predizem a classe correta (*Palythoa caribaeorum*) como a segunda provável.

A Figura 5.5 apresenta os resultados do LIME para as duas classes prováveis dos 3 modelos utilizando a Figura 5.2(c). Um ponto interessante dessa imagem é a falta do coral centralizado e a presença de texto na imagem. Observa-se que os 3 modelos interagem com os textos presentes na imagem avaliando positiva ou negativamente. Assim sendo, os resultados para a imagem não são confiáveis, pois existe a interação do modelo com outros artefatos da imagem. Consequentemente, torna-se necessário a limpeza de textos, marca d'água e outros ruídos para uma predição mais confiável do modelo.

5.1.2 Experimento com sub-imagens

Os resultados para as sub-imagens com o limiar de 224 × 244 estão apresentados na Tabela 5.5. Nessa tabela podemos observar que os melhores resultados para a base de validação ficaram com a combinação entre os modelos e, para a base de teste, com o modelo EfficientNetB0 + regressão logística.



(a) (b) (c)

(d) (e) (f)

Figura 5.4: Resultados do LIME para Figura 5.2(b) utilizando as duas classes mais prováveis dos modelos. Os pontos em verde e vermelho indicam que a região influência positivamente e negativamente, respectivamente, na predição do modelo para a classe avaliada. (a) EfficientNetB7 + regressão logística - Classe 1 (b) ResNet101 (ImageNet) - Classe 1 (c) ResNet (PLC) - Classe 1 (d) EfficientNetB7 + regressão logística - Classe 2 (e) ResNet101 (ImageNet) - Classe 2 (c) ResNet (PLC) - Classe 2

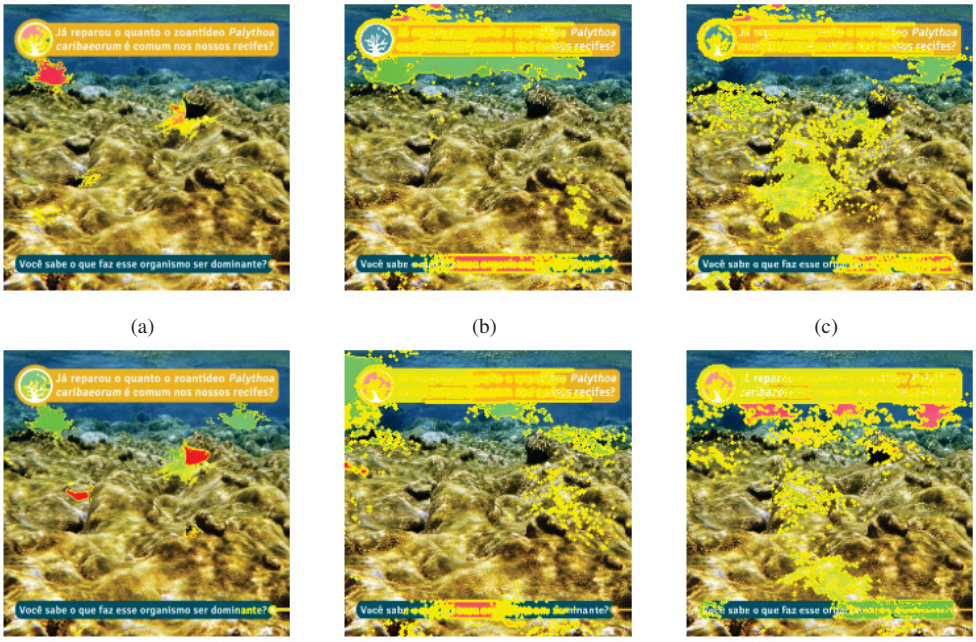
Tabela 5.4: Probabilidades preditas pelos modelos para a Figura 5.2(c). Classe com maior probabilidade (classe 1), segunda classe com maior probabilidade (classe 2)

	Classe 1		**Classe 2**	
	Classe	Prob.	Classe	Prob.
EfficientNetB7 + regressão logística	*Siderastrea stellata*	0,71	*Palythoa caribaeorum*	0,12
ResNet101 (ImageNet)	*Porites astreoides*	0,52	*Siderastrea stellata*	0,22
ResNet101 (PLC)	*Porites astreoides*	0,55	*Palythoa caribaeorum*	0,22

Tabela 5.5: Resultados encontrados no experimento utilizando as sub-imagens 224 × 224 para as bases de validação e teste

	Acurácia		**F1**		**MCC**	
	Val.	Teste	Val.	Teste	Val.	Teste
EfficientNetB0 + regressão logística	0,80	**0,79**	0,76	**0,80**	**0,77**	**0,73**
ResNet101 (ImageNet)	0,78	0,71	0,75	0,70	0,73	0,62
ResNet101 (PLC)	0,73	0,74	0,71	0,73	0,69	0,66
Combinação	**0,81**	0,77	**0,80**	0,79	**0,77**	0,70

Outro ponto a se destacar em relação aos resultados apresentados na Tabela 5.5 é a diferença entre as ResNet101. A ResNet101 (ImageNet) possui melhores resultados na base de validação em relação a inicializada com os pesos da base PLC. Porém, o desempenho da ResNet101 (ImageNet) cai significativamente para a base de teste, enquanto a ResNet101 (PLC) mantém o seu desempenho entre as bases. Portanto, a utilização da base PLC pode ter gerado um aumento da generalização do modelo.

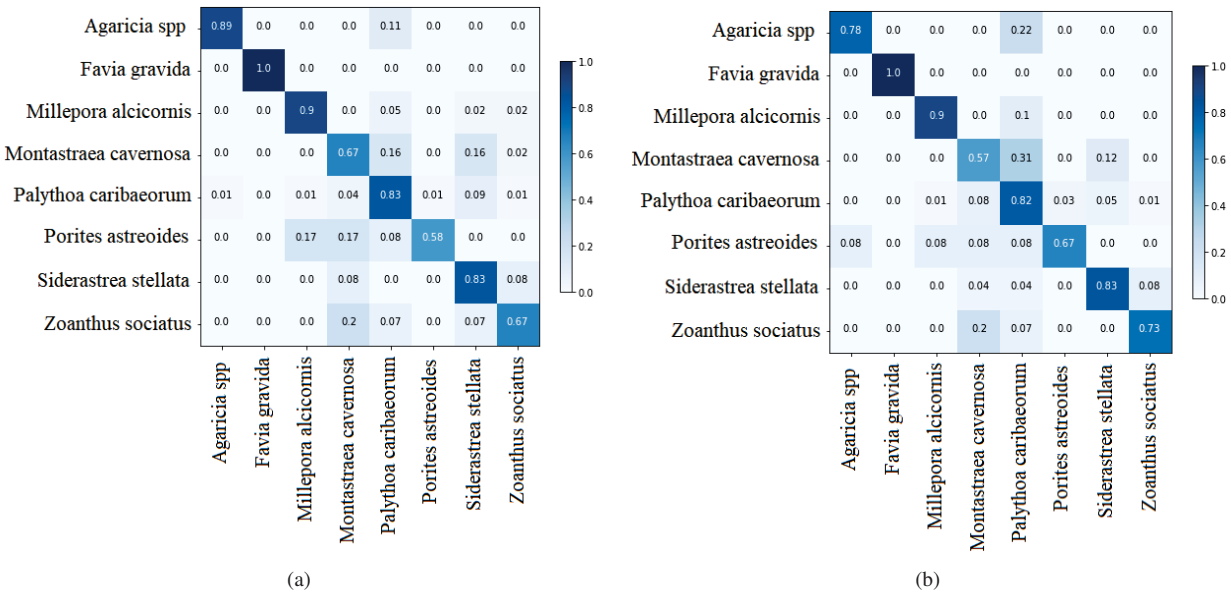


(a) (b) (c)

(d) (e) (f)

Figura 5.5: Resultados do LIME para Figura 5.2(c) utilizando as duas classes mais prováveis dos modelos. Os pontos em verde e vermelho indicam que a região influência positivamente e negativamente, respectivamente, na predição do modelo para a classe avaliada. (a) EfficientNetB7 + regressão logística - Classe 1 (b) ResNet101 (ImageNet) - Classe 1 (c) ResNet (PLC) - Classe 1 (d) EfficientNetB7 + regressão logística - Classe 2 (e) ResNet101 (ImageNet) - Classe 2 (c) ResNet (PLC) - Classe 2

A Figura 5.6 apresenta as matrizes de confusão para os modelos EfficientNetB0 + regressão logística e o da combinação. Observa-se que as matrizes de confusão para ambos os modelos são semelhantes.



(a) (b)

Figura 5.6: Matriz de confusão dos modelos (a) EfficientNetB0 + regressão logística e (b) combinação para a base de teste das sub-imagens 224 × 224

Comparando com as imagens inteiras, o desempenho na base de validação foi parecido mesmo considerando que a base das sub-imagens possui uma classe a mais. Contudo, quando se

compara os resultados na base de teste, o desempenho entre as metodologias é de cerca de 10% melhor para as sub-imagens.

Esses resultados mostram que, para alguns casos, as informações extras na imagem influenciam negativamente a predição do modelo. Outro ponto é que o maior número de imagens também auxilia no aumento da generalização dos modelos.

Na Tabela 5.6 estão apresentados os resultados para as sub-imagens com o limiar de 128 × 128. Os resultados são semelhantes entre as duas metodologias de treinamento baseadas na MobileNetV2. Entretanto, o treinamento completo do modelo possui uma pequena vantagem em todas as métricas analisadas. A Figura 5.7 apresenta as matrizes de confusão para o modelo MobileNetV2.

Tabela 5.6: Resultados encontrados no experimento utilizando as sub-imagens 128 × 128 para as bases de validação e teste

	Acurácia		**F1**		**MCC**	
	Val.	Teste	Val.	Teste	Val.	Teste
MobileNetV2 + regressão logística	0,65	0,65	0,58	0,58	0,60	0,58
MobileNetV2	**0,66**	**0,67**	**0,59**	**0,64**	**0,61**	**0,60**

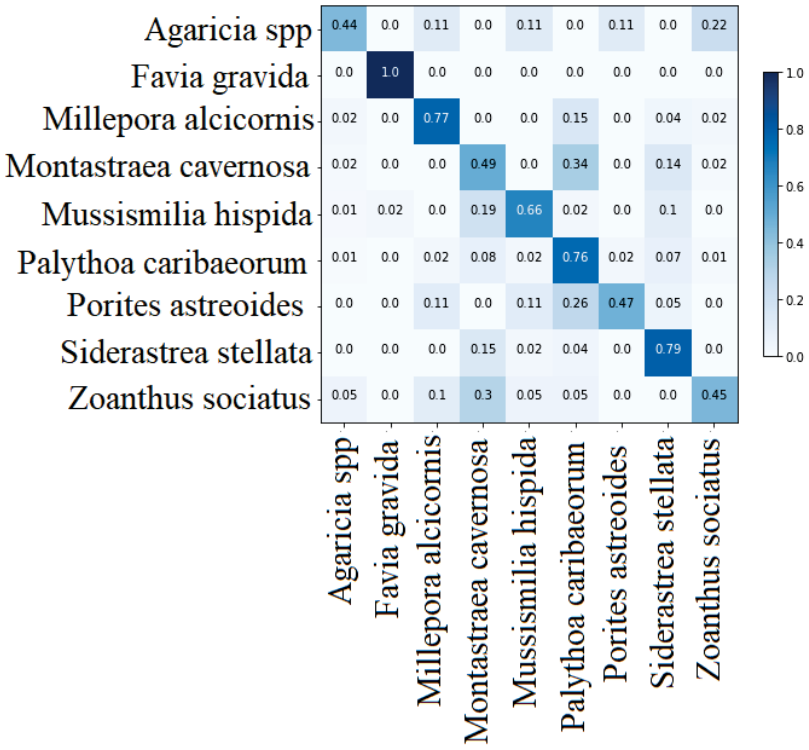


Figura 5.7: Matriz de Confusão do modelo MobileNetV2 para a base de teste das sub-imagens 128 × 128

Os testes com as sub-imagens com limiar de 128 × 128 demonstraram um desempenho inferior na base de validação. Porém, como nas imagens com limiar de 224 × 224 não há uma diminuição do desempenho significativa para a base de teste. Mostrando que a utilização de sub-imagens impacta positivamente na generalização do modelo em contraste com os resultados apresentados com as imagens inteiras.

Um ponto a se destacar em relação à matriz de confusão é o baixo desempenho do modelo para as classes *Agaricia spp*, *Montastraea cavernosa* e *Zoanthus sociatus*. Esse baixo

desempenho pode ter sido ocasionado devido à qualidade das sub-imagens geradas com o limiar de 128 × 128, gerando sub-imagens pouco representativas.

5.2 SEGMENTAÇÃO SEMÂNTICA E LOCALIZAÇÃO DE OBJETOS

A Tabela 5.7 apresenta os resultados para a segmentação semântica binária utilizando a U-net (Pix2Pix). Observa-se a existência de uma boa generalização por parte do modelo com resultados parecidos na base de validação e de teste.

Tabela 5.7: Resultados encontrados na tarefa de segmentação semântica binária utilizando a U-net (Pix2Pix) para as bases de validação e teste

	Acurácia por pixel		**mIoU**	
	Val.	Teste	Val.	Teste
U-net (Pix2Pix)	0,86	0,86	0,74	0,70

A Figura 5.8 apresenta 5 exemplos da base de teste que mostra a imagem original, seu mapa de segmentação e o mapa de segmentação predito pelo modelo. Os exemplos mostram que o modelo consegue ter boa localização dos corais na imagem em diferentes situações.

Em projetos futuros utilizando a base de dados pode-se avaliar modelos mais robustos que conseguem utilizar imagens com resolução superior a 128 × 128. Também é possível avaliar modelos capazes de realizar a tarefa de segmentação semântica categórica.

A Tabela 5.8 apresenta os resultados para a localização de objetos binária utilizando a Yolov5. Entretanto, diferente da tarefa de segmentação, a Yolov5 apresentou resultados inferiores para a base de teste. Em projetos futuros pode-se utilizar outros modelos ou diferentes configurações dos parâmetros de treinamento para mitigar essa baixa generalização.

Tabela 5.8: Resultados encontrados na tarefa de localização de objetos binária utilizando a Yolov5 para as bases de validação e teste

	Precisão		***Recall***		**AP50**		**AP95**	
	Val.	Teste	Val.	Teste	Val.	Teste	Val.	Teste
Yolov5	0,72	0,58	0,48	0,40	0,55	0,41	0,38	0,27

A Figura 5.9 mostra exemplos de predições na base de teste. Observa-se que o modelo consegue localizar os corais centralizados na imagem e em alguns casos corais de difícil identificação por serem relativamente pequenos em comparação com o tamanho da imagem ou estarem nas bordas da imagem.

Esses resultados iniciais mostram que base pode ser utilizada para outros trabalhos com a finalidade de aperfeiçoar esse desempenho para a localização de objeto binária ou desenvolver projetos com a localização de objetos categórica.

Os exemplos mostram ser possível a utilização do mapa de segmentação para criação das *bounding boxes*. Entretanto, em alguns casos se gerou *bounding boxes* separadas para corais próximos e em outros casos gerou-se uma única *bounding box*. Essa falta de padronização dificulta o treinamento e predição do modelo.

5.3 ANÁLISE DOS RESULTADOS EXPERIMENTAIS

Em relação aos experimentos utilizando as imagens inteiras, a solução de se utilizar a EfficientNetB7 como extrator de características e a regressão logística como classificador obteve

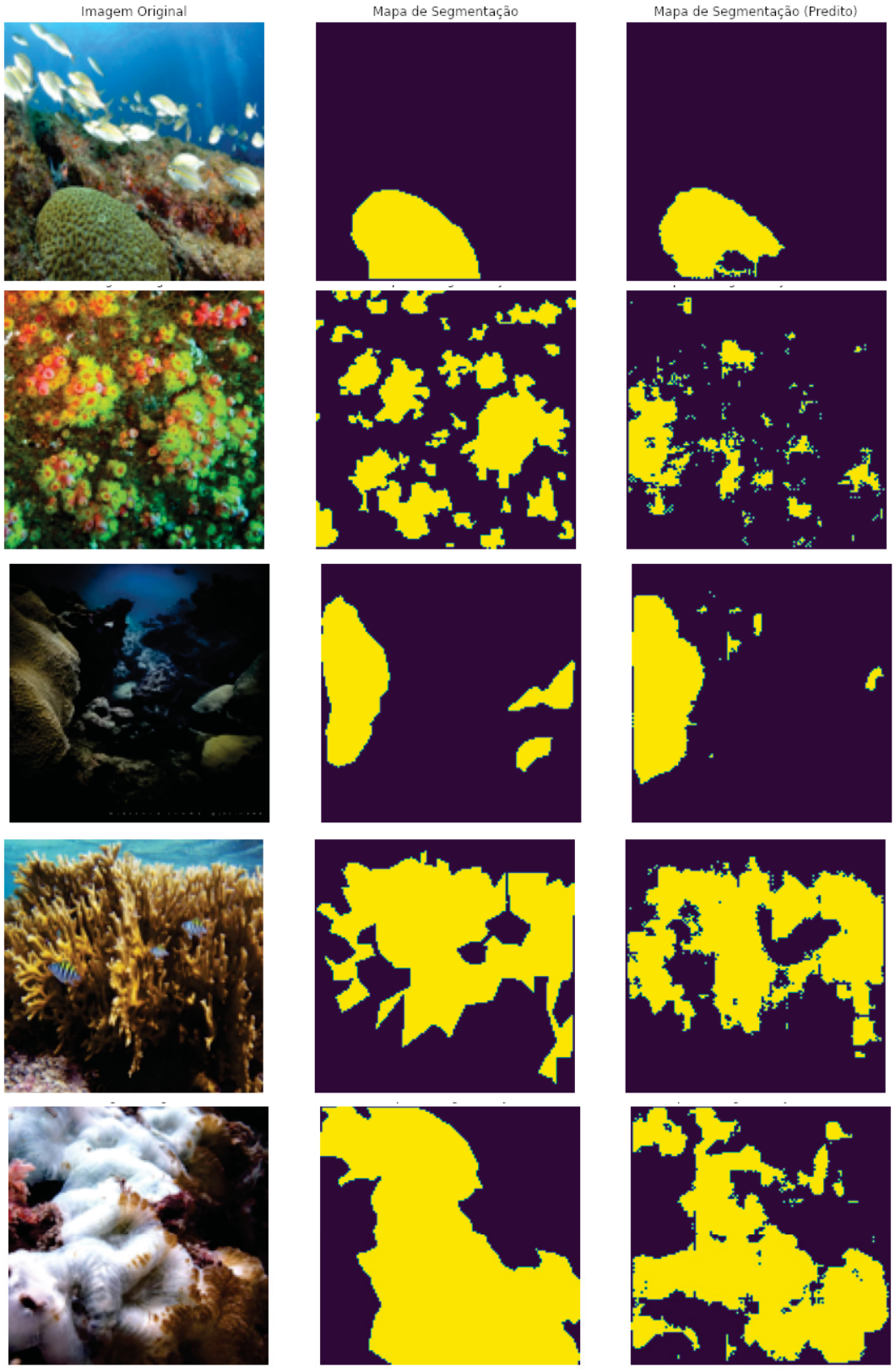


Figura 5.8: Cinco mapas de segmentação preditos pela U-net (Pix2Pix) para a base de teste

(a)

(b)

Figura 5.9: 16 exemplos de predições da Yolov5 para a base de teste - (a) rótulos (b) predição

resultados superiores que a ResNet101 treinadas da forma *end-to-end*. O modelo (EfficientNetB7 + regressão logística) obteve resultados em torno de 0,8 nas métricas para a base de validação. Para a base de teste, o modelo obteve em torno de 0,7 para a acurácia e F1, para a métrica MCC o resultado foi de 0,6.

Os resultados com LIME demonstraram que as predições dos modelos utilizados não foram aleatórias. Mostrou-se que os modelos ativam para regiões onde existe o coral na imagem. Porém, os resultados mostraram que a classificação também é influenciada por textos presentes nas imagens. Em projetos futuros deve-se avaliar a possibilidade de remoção desses artefatos a fim de gerar predições mais confiáveis.

Os resultados apresentados para as sub-imagens com limiar de 224 × 224 mostram que a configuração EfficientNetB0 com a regressão logística continuou com os melhores resultados. Entretanto, a ResNet101 (ImageNet) nesses experimentos obteve resultados mais próximos dos obtidos com a EfficientNet + regressão logística. A proximidade dos resultados entre os modelos fez com que a combinação entre eles em algumas métricas obtivesse os melhores resultados.

Nos experimentos com as sub-imagens com limiar 224 x 224 também observou-se que a ResNet101 (PLC) obteve resultados melhores na base de teste que na base de validação para as métricas de Acurácia e F1. Consequentemente, superando os resultados encontrados pela ResNet101 (ImageNet) na base de teste.

Em geral, os resultados do experimento com as sub-imagens 224 × 224 foram semelhantes aos das imagens inteiras para base de validação, porém melhores para a base de teste. Mostrando que a utilização das sub-imagens aumenta a generalização dos modelos, devido à redução dos ruídos e do aumento do número de imagens. Para os testes com o limiar 128 × 128 demonstrou-se que utilizar uma resolução baixa para entrada dos modelos reduziu os resultados para em torno de 0,6 nas bases de validação e teste.

Os testes iniciais para segmentação semântica binária e localização de objetos binária mostram que pode-se utilizar esses modelos para uma base de corais. Em relação à segmentação semântica, obteve-se resultados de mIoU em torno de 0,7. Contudo, existe a possibilidade da utilização de outros modelos mais robustos que criam mapas de segmentação com resolução superiores a 128 × 128 pixels.

Acerca da localização de objetos, pode-se utilizar um modelo que recebe e gera imagens com resolução de 640 × 640. Todavia, seus resultados foram medianos. Os resultados devem ser ponderados, pois os rótulos utilizados foram extraídos do mapa de segmentação. Portanto, não se tratava de uma rotulação nativa para a tarefa. Consequentemente, foram gerados pequenos objetos no *ground truth* que poderiam estar agrupados em uma *bounding box* única.

A Yolov5 também apresentou resultados inferiores para a base de teste, mostrando assim uma baixa generalização. Esses resultados podem ser ajustados utilizando novos parâmetros de treinamento ou com o emprego de outros modelos para a realização da tarefa de localização de objetos.

6 CONCLUSÃO

Nesta dissertação tratamos do problema de classificação e segmentação de corais utilizando aprendizagem de máquina. Através dos resultados apresentados, podemos concluir que o trabalho responde de forma satisfatória às perguntas de pesquisa apresentadas na Seção 1.2.

O trabalho conseguiu formar uma base de dados[1] de corais nacionais com a colaboração do LECOM da UFRN. A base de dados possui uma metodologia clara para as separações da base entre treinamento, validação e teste, assim podendo servir como referência para outros projetos que desejarem utilizar a base de dados para novos experimentos.

Em relação aos resultados experimentais, um dos desafios elencados para o trabalho era do número de imagens disponíveis na base de dados. Os resultados mostraram que os modelos de aprendizagem de máquina apresentam dificuldades em relação ao número de imagens presentes na base.

Para os experimentos com as imagens inteiras existe uma queda de cerca de 10% nas métricas analisadas da base de validação para a base de teste. Entretanto, em relação as sub-imagens esse comportamento não acontece devido ao menor ruído nas imagens e o maior número de imagens presentes na base.

Em relação à tarefa de classificação, a solução de se utilizar a EfficientNet como extrator de características e a regressão logística como classificador obteve resultados superiores que a ResNet101 treinadas da forma *end-to-end* para as imagens inteiras e para as sub-imagens 224 x 224. Para as sub-imagens 128 x 128 os resultados entre os modelos MobileNetV2 + regressão logística e a MobileNetV2 foram semelhantes.

Os resultados para o LIME constataram que os modelos de classificação utilizam os corais como regiões relevantes para as suas predições. Entretanto, os resultados mostraram que os modelos também sofrem influência de artefatos adicionados pelos usuários na rede social, como caixas de texto. Esses artefatos podem confundir o modelo e gerar predições não confiáveis.

O trabalho também apresentou resultados iniciais para tarefas mais complexas (segmentação semântica e localização de objetos) que necessitam da localização dos corais nas imagens. Para a tarefa de segmentação semântica esses são os primeiros resultados a utilizarem um mapa de segmentação real e não uma aproximação. Os experimentos iniciais para ambas as tarefas apresentaram resultados promissores que podem ser melhorados em projetos futuros. Esses modelos binários podem ser utilizados para filtrar imagens realizadas por mergulhadores com a finalidade de separar as imagens que podem ou não possuir corais, assim reduzindo o tempo empregado na seleção das imagens coletadas.

6.1 POSSÍVEIS TRABALHOS FUTUROS

O trabalho apresentou resultados para três tarefas de visão computacional utilizando diferentes metodologias de treinamento de modelos, além de desenvolver uma base de dados que pode se utilizada para as principais tarefas de visão computacional.

Durante as etapas de desenvolvimento da dissertação discutiu-se pontos que podem servir para futuras dissertações visando enriquecer os resultados encontrados nesse trabalho. A proposta de possíveis adições e modificações no atual trabalho são:

- Aprimorar o desempenho da tarefa de classificação utilizando outros modelos

[1] https://doi.org/10.5281/zenodo.7338208

- Utilizar *Generative Adversarial Networks* (GANs) para criar artificialmente novas imagens para a base de dados
- Utilizar técnicas de *upscaling* para se utilizar sub-imagens de corais com resolução inferior a 224 × 224
- Aprofundar os estudos para segmentação semântica e localização binária e realizar experimentos com os rótulos categóricos
- Utilizar modelos de segmentação semântica binária que criam mapas de segmentação com resoluções superiores a 128 x 128
- Realizar testes iniciais para os rótulos binários e categóricos para a tarefa de segmentação instanciada

Por fim, os resultados já apresentados nesse trabalho podem ser utilizados para o desenvolvimento de uma ferramenta, que o usuário a utilizaria para realizar as tarefas de classificação, segmentação ou localização de corais. Entretanto, com a adição de uma função colaborativa para que o usuário possa corrigir as predições do modelo, com isso aumentando o número de imagens rotuladas. Por fim, os modelos podem ser treinados novamente com a nova base de dados a fim de melhor o seu desempenho ou serem treinados constantemente usando *self-learning*.

REFERÊNCIAS

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

Academy, D. S. (2022). Deep learning book. `https://www.deeplearningbook.com.br`. Acessado em 09/08/2022.

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P. e Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282.

Alonso, I., Cambra, A., Munoz, A., Treibitz, T. e Murillo, A. C. (2017). Coral-segmentation: Training dense labeling models with sparse ground truth. Em *2017 IEEE International Conference on Computer Vision Workshop (ICCVW)*, páginas 2874–2882, Los Alamitos, CA, USA. IEEE Computer Society.

Badrinarayanan, V., Kendall, A. e Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495.

Beijbom, O., Edmunds, P. J., Kline, D. I., Mitchell, B. G. e Kriegman, D. (2012). Automated annotation of coral reef survey images. Em *2012 IEEE Conference on Computer Vision and Pattern Recognition*, páginas 1170–1177.

Beijbom, O., Edmunds, P. J., Roelfsema, C., Smith, J., Kline, D. I., Neal, B. P., Dunlap, M. J., Moriarty, V., Fan, T.-Y., Tan, C.-J., Chan, S., Treibitz, T., Gamst, A., Mitchell, B. G. e Kriegman, D. (2015). Towards automated annotation of benthic survey images: Variability of human experts and operational modes of automation. *PLOS ONE*, 10(7):1–22.

Beijbom, O., Treibitz, T., Kline, D., Eyal, G., Khen, A., Neal, B., Loya, Y., Mitchell, B. e Kriegman, D. (2016). Improving automated annotation of benthic survey images using wide-band fluorescence. *Scientific Reports*, 6:23166.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B. e Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. Em *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, páginas 108–122.

CAUCHY, A. (1847). Methode generale pour la resolution des systemes d'equations simultanees. *C.R. Acad. Sci. Paris*, 25:536–538.

Chen, L., Papandreou, G., Kokkinos, I., Murphy, K. e Yuille, A. L. (2016). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915.

Chen, L., Papandreou, G., Schroff, F. e Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587.

Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F. e Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11211 LNCS:833–851.

Chen, P.-Y., Chen, C.-C., Chu, L. e McCarl, B. (2015). Evaluating the economic damage of climate change on global coral reefs. *Global Environmental Change*, 30:12 – 20.

Chen, T. e Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754.

Chicco, D. e Jurman, G. (2020). The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(1):6.

Chollet, F. (2016). Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S. e Schiele, B. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem:3213–3223.

Cortes, C. e Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, 20(3):273–297.

CS231n (2020). Cs231n course materials. `https://cs231n.github.io/`. Acessado em 24/02/2021.

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H. e Wei, Y. (2017). Deformable convolutional networks. *CoRR*, abs/1703.06211.

den Bergh, M. V., Boix, X., Roig, G. e Gool, L. V. (2013). SEEDS: superpixels extracted via energy-driven sampling. *CoRR*, abs/1309.3848.

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

Duchi, J., Hazan, E. e Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Efron, B., Hastie, T., Johnstone, I. e Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, 32(2):407 – 499.

Elson, J., Douceur, J. J., Howell, J. e Saul, J. (2007). Asirra: A captcha that exploits interest-aligned manual image categorization. Em *Proceedings of 14th ACM Conference on Computer and Communications Security (CCS)*. Association for Computing Machinery, Inc.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J. e Zisserman, A. (2012). The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-Gonzalez, P. e Garcia-Rodriguez, J. (2018). A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing Journal*, 70:41–65.

Glorot, X. e Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. Em Teh, Y. W. e Titterington, M., editores, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 de *Proceedings of Machine Learning Research*, páginas 249–256, Chia Laguna Resort, Sardinia, Italy. PMLR.

Goodfellow, I., Bengio, Y. e Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Goodfellow, I. J., Shlens, J. e Szegedy, C. (2015). Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572.

Graf, A., Koch-Kramer, A. e Lindqvist, L. (2021). Instaloader. `https://github.com/instaloader/instaloader`. Acessado em 09/08/2022.

Gómez-Ríos, A., Tabik, S., Luengo, J., Shihavuddin, A., Krawczyk, B. e Herrera, F. (2019). Towards highly accurate coral texture images classification using deep convolutional neural networks and data augmentation. *Expert Systems with Applications*, 118:315 – 328.

He, K., Zhang, X., Ren, S. e Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

Hinton, G. (2012). Neural networks for machine learning. Coursera, video lectures.

Hoegh-Guldberg, O. (1999). Climate change, coral bleaching and the future of the world's coral reefs. *Marine and Freshwater Research*, 50.

Hoegh-Guldberg, O. (2011). Coral reef ecosystems and anthropogenic climate change. *Regional Environmental Change*, 11:215–227.

Hoey, A. S., Howells, E., Johansen, J. L., Hobbs, J.-P. A., Messmer, V., McCowan, D. M., Wilson, S. K. e Pratchett, M. S. (2016). Recent advances in understanding the effects of climate change on coral reefs. *Diversity*, 8(2).

Huang, G., Liu, Z. e Weinberger, K. Q. (2016). Densely connected convolutional networks. *CoRR*, abs/1608.06993.

Hughes, T. P., Anderson, K. D., Connolly, S. R., Heron, S. F., Kerry, J. T., Lough, J. M., Baird, A. H., Baum, J. K., Berumen, M. L., Bridge, T. C., Claar, D. C., Eakin, C. M., Gilmour, J. P., Graham, N. A. J., Harrison, H., Hobbs, J.-P. A., Hoey, A. S., Hoogenboom, M., Lowe, R. J., McCulloch, M. T., Pandolfi, J. M., Pratchett, M., Schoepf, V., Torda, G. e Wilson, S. K. (2018). Spatial and temporal patterns of mass bleaching of corals in the anthropocene. *Science*, 359(6371):80–83.

Ioffe, S. e Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Isola, P., Zhu, J.-Y., Zhou, T. e Efros, A. A. (2016). Image-to-image translation with conditional adversarial networks.

Jocher, G. (2020). Yolov5. `https://github.com/ultralytics/yolov5`. Acessado em 09/08/2022.

King, A., Bhandarkar, S. M. e Hopkinson, B. M. (2018). A comparison of deep learning methods for semantic segmentation of coral reef survey images. Em *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, páginas 1475–14758.

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

Krizhevsky, A., Sutskever, I. e Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Em Pereira, F., Burges, C. J. C., Bottou, L. e Weinberger, K. Q., editores, *Advances in Neural Information Processing Systems 25*, páginas 1097–1105. Curran Associates, Inc.

Leibniz, G. W. (1676). Memoir using the chain rule (citado em TMME 7:2&3 p 321-332, 2010).

L'Hôpital, G. F. A. (1696). *Analyse des infiniment petits, pour l'intelligence des lignes courbes*. L'Imprimerie Royale, Paris.

Lin, G., Milan, A., Shen, C. e Reid, I. D. (2016). Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *CoRR*, abs/1611.06612.

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P. e Zitnick, C. L. (2014). Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312.

Liu, L., Ouyang, W., Wang, X., Fieguth, P. W., Chen, J., Liu, X. e Pietikäinen, M. (2018). Deep learning for generic object detection: A survey. *CoRR*, abs/1809.02165.

Long, J., Shelhamer, E. e Darrell, T. (2014). Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038.

Lumini, A., Nanni, L. e Maguolo, G. (2019). Deep learning for plankton and coral classification.

Lundberg, S. M. e Lee, S.-I. (2017). A unified approach to interpreting model predictions. Em Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S. e Garnett, R., editores, *Advances in Neural Information Processing Systems 30*, páginas 4765–4774. Curran Associates, Inc.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D. e Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083.

Mahapatra, S. (2018). Why deep learning over traditional machine learning? `https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063`. Acessado em 24/02/2021.

Mahmood, A., Bennamoun, M., An, S. e Sohel, F. A. (2016). Resfeats: Residual network based features for image classification. *CoRR*, abs/1611.06656.

Malik, M., Malik, M. K., Mehmood, K. e Makhdoom, I. (2020). Automatic speech recognition: a survey. *Multimedia Tools and Applications*.

Milletari, F., Navab, N. e Ahmadi, S. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation. *CoRR*, abs/1606.04797.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, New York.

Moore, B. E. e Corso, J. J. (2020). Fiftyone. *GitHub*. Acessado em 09/08/2022.

Oberweger, M., Wohlhart, P. e Lepetit, V. (2015). Hands deep in deep learning for hand pose estimation. *CoRR*, abs/1502.06807.

Oquab, M., Bottou, L., Laptev, I. e Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Polyak, B. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17.

Ribeiro, M. T., Singh, S. e Guestrin, C. (2016). "why should I trust you?": Explaining the predictions of any classifier. *CoRR*, abs/1602.04938.

Rodriguez, D. (2020). multilayer-perceptron. `https://github.com/d-r-e/multilayer-perceptron`. Acessado em 09/08/2022.

Ronneberger, O., Fischer, P. e Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

Rumelhart, D. E., Hinton, G. E. e Williams, R. J. (1986). Learning Representations by Back-propagating Errors. *Nature*, 323(6088):533–536.

Sandler, M., Howard, A. G., Zhu, M., Zhmoginov, A. e Chen, L. (2018). Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *CoRR*, abs/1801.04381.

SHAP (2021). Shap documentation. `https://github.com/slundberg/shap`. Acessado em 24/05/2021.

Shihavuddin, A. (2017). Coral reef dataset. `https://data.mendeley.com/datasets/86y667257h/2`.

Shihavuddin, A., Gracias, N., Garcia, R., Gleason, A. C. R. e Gintert, B. (2013). Image-based coral reef classification and thematic mapping. *Remote Sensing*, 5(4):1809–1841.

Simonyan, K. e Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. e Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Szegedy, C., Ioffe, S. e Vanhoucke, V. (2016). Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V. e Rabinovich, A. (2014). Going deeper with convolutions. *CoRR*, abs/1409.4842.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. e Wojna, Z. (2015). Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567.

Tan, M. e Le, Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946.

TensorFlow (2020). U-net (pix2pix). `https://www.tensorflow.org/tutorials/images/segmentation`. Acessado em 09/08/2022.

TensorFlow (2021). Cats and dogs. `https://www.tensorflow.org/datasets/catalog/cats_vs_dogs`. Acessado em 24/02/2021.

Tin Kam Ho (1995). Random decision forests. Em *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, páginas 278–282 vol.1.

Torfi, A., Shirvani, R. A., Keneshloo, Y., Tavaf, N. e Fox, E. A. (2020). Natural language processing advancements by deep learning: A survey.

Wada, K. (2016). labelme: Image Polygonal Annotation with Python. `https://github.com/wkentaro/labelme`.

Wang, M. e Deng, W. (2018). Deep face recognition: A survey. *CoRR*, abs/1804.06655.

Williams, I. D., Couch, C. S., Beijbom, O., Oliver, T. A., Vargas-Angel, B., Schumacher, B. D. e Brainard, R. E. (2019). Leveraging automated image analysis tools to transform our capacity to assess status and trends of coral reefs. *Frontiers in Marine Science*, 6:222.

Wilson, D. R. e Martinez, T. R. (2003). The general inefficiency of batch training for gradient descent learning. *Neural Networks*, 16(10):1429–1451.

Xu, L., Bennamoun, M., Boussaid, F., An, S. e Sohel, F. (2019). Coral classification using densenet and cross-modality transfer learning. Em *2019 International Joint Conference on Neural Networks (IJCNN)*, páginas 1–8.

Yakubovskiy, P. (2019). Segmentation models. `https://github.com/qubvel/segmentation_models`.

Yosinski, J., Clune, J., Bengio, Y. e Lipson, H. (2014). How transferable are features in deep neural networks? *CoRR*, abs/1411.1792.

Yu, F. e Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions.

Zhang, Z., Liu, Q. e Wang, Y. (2017). Road extraction by deep residual u-net. *CoRR*, abs/1711.10684.

Zhang, Z., Zhang, X., Peng, C., Cheng, D. e Sun, J. (2018). Exfuse: Enhancing feature fusion for semantic segmentation. *CoRR*, abs/1804.03821.

Zhao, H., Shi, J., Qi, X., Wang, X. e Jia, J. (2016). Pyramid scene parsing network. *CoRR*, abs/1612.01105.

Zoph, B., Vasudevan, V., Shlens, J. e Le, Q. V. (2017). Learning transferable architectures for scalable image recognition. *CoRR*, abs/1707.07012.

APÊNDICE A – PLC

O anexo demonstra a distribuição da base PLC (Seção A.1) utilizada para relizar-se o *transfer learning* utilizando a ResNet101 e os resultados do treinamento do modelo (Seção A.2).

A.1 DISTRIBUIÇÃO DA BASE

A Tabela A.1 mostra a distribuição da base utilizada para o treinamento do modelo. Selecionou-se 12 classes das 20 presentes na base e retirou-se sub-imagens de 224 × 224 centradas no píxel rotulado. Foi selecionado as imagens do *reference set* como treinamento e as imagens do *evaluation set* com os rótulos *archived* como validação.

Tabela A.1: Distribuição da base de dados PLC utilizando sub-imagens 224 x 224 e 12 classes

Classes	**Treino**	**Validação**
1- *Acropora*	5181	159
2- *Favia*	1343	74
3- *Favites*	1124	55
4- *Macroalgae*	22783	700
5- *Millepora*	3114	158
6- *Montipora*	12082	444
7- *Other scleractinians*	21614	724
8- *Pavona*	2033	52
9- *Platygyra*	703	44
10- *Pocillopora*	9756	244
11- *Porites*	11400	332
12- *Soft Coral*	4860	144
Total	**95993**	**3130**

A Figura A.1 apresenta a distribuição da base PLC ordenada por número de imagens. Observa-se que existe um desbalanceamento na base com algumas classes mais predominantes que as outras.

A.2 RESULTADOS

A ResNet101 foi treinada com os parâmetros apresentados na Tabela A.2.

Tabela A.2: Parâmetros de treinamento da ResNet101 para a base PLC

	SGD			**Modelo**		
Etapas de Treinamento	Taxa de Aprendizado	*Momentum*	Decaimento	Batch size	Épocas	Dropout
Camada Densa	10^{-4}	0,9	0,005	32	43	0,2
CNN Completa	10^{-5}	0,9	0,020	32	59	0,2

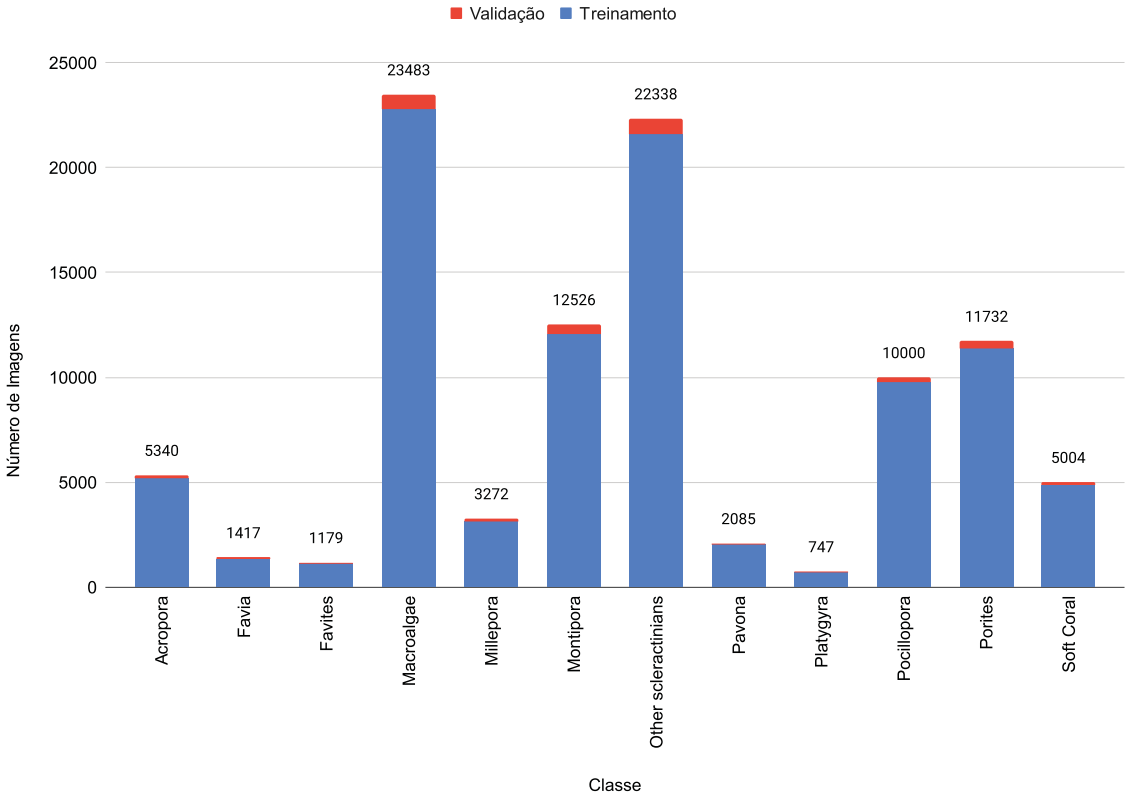


Figura A.1: Distribuição da base de dados (PLC) utilizando sub-imagens 224 x 224 e 12 classes ordenada por número de imagens totais

Os resultados encontrados para a base de validação estão apresentados na Tabela A.3 e a matriz de confusão está apresentada na Figura A.2.

Tabela A.3: Resultados para a base PLC na base de validação utilizando a ResNet101

	Acurácia	**F1**	**MCC**
ResNet101	0,85	0,78	0,82

Observa-se que o modelo para base PLC tem bons resultados com essas 12 classes. Ele tem uma leve tendência para predizer a classe *Other scleractinians*. Esse comportamento pode ser função de dois fatores: a classe ser majoritária e a não homogeneidade das imagens por se tratar de uma classe que agrupa várias espécies. As classes com os piores desempenhos para o modelo foram: *Favites*, *Favia*, *Pavona*, *Platygyra*.

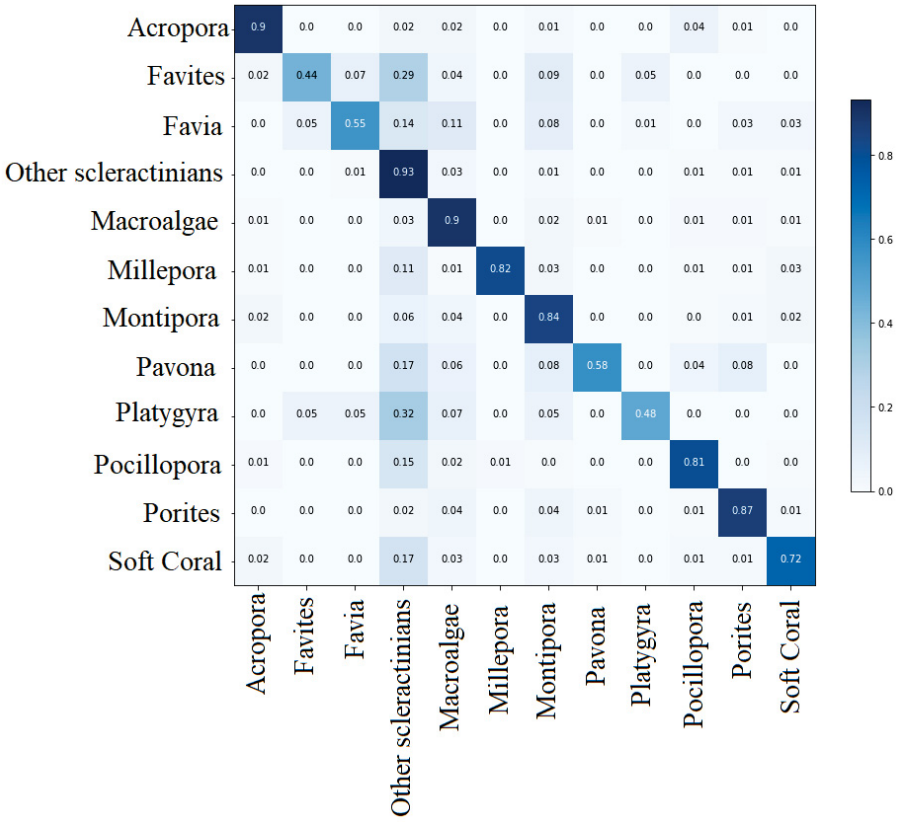


Figura A.2: Matriz de confusão da ResNet101 na base de validação da PLC