UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

UM ESTUDO SOBRE ABORDAGENS PARA AVALIAÇÃO *OUT-OF-SAMPLE* DE MODELOS DE CLASSIFICAÇÃO DE ANIMAIS EM IMAGENS DE ARMADILHAS FOTOGRÁFICAS

FRANCISCO FAGNER DO REGO CUNHA

UM ESTUDO SOBRE ABORDAGENS PARA AVALIAÇÃO *OUT-OF-SAMPLE* DE MODELOS DE CLASSIFICAÇÃO DE ANIMAIS EM IMAGENS DE ARMADILHAS FOTOGRÁFICAS

Dissertação apresentada ao Programa de Pós-Graduação em Informática do Instituto de Computação da Universidade Federal do Amazonas, Campus Universitário Senador Arthur Virgílio Filho, como requisito parcial para a obtenção do grau de Mestre em Informática.

Orientador: Eulanda Miranda dos Santos
Coorientador: Juan Gabriel Colonna

Manaus - AM

Abril de 2019

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Cunha, Francisco Fagner do Rego
C972u Um Estudo sobre Abordagens para Avaliação Out-of-sample de Modelos de Classificação de Animais em Imagens de Armadilhas Fotográficas / Francisco Fagner do Rego Cunha. 2019
78 f.: il. color; 31 cm.

Orientadora: Eulanda Miranda dos Santos
Coorientador: Juan Gabriel Colonna
Dissertação (Mestrado em Informática) - Universidade Federal do Amazonas.

1. Redes neurais profundas. 2. Armadilhas fotográficas. 3. Aprendizagem de máquina. 4. Particionamento dos dados. 5. Monitoramento da vida selvagem. I. Santos, Eulanda Miranda dos II. Universidade Federal do Amazonas III. Título

FOLHA DE APROVAÇÃO

"UM ESTUDO SOBRE ABORDAGENS PARA AVALIAÇÃO OUT-OF-SAMPLE DE MODELOS DE CLASSIFICAÇÃO DE ANIMAIS EM IMAGENS DE ARMADILHAS FOTOGRÁFICAS"

FRANCISCO FAGNER DO REGO CUNHA

Dissertação de Mestrado defendida e aprovada pela banca examinadora constituída pelos Professores:

Prof. Marco Antônio Pinheiro de Cristo - PRESIDENTE

Prof. José Reginaldo Hughes Carvalho - MEMBRO INTERNO

Prof. André Luiz da Costa Carvalho - MEMBRO EXTERNO

Manaus, 01 de Abril de 2019

Agradecimentos

Agradeço primeiramente ao Supremo Arquiteto do Universo, que é Deus, por guiar meus passos no caminho da retidão. É n'Ele em quem deposito minha confiança nos extremos lances da vida.

Agradeço à minha mãe, Maria Zuleide do Rego Cunha, e ao meu pai, Florêncio Joviniano Cunha Neto, que me educaram e me deram as condições necessárias para continuar estudando.

Agradeço à minha orientadora Prof^{a}. Dr^{a}. Eulanda Miranda dos Santos e ao meu coorientador Prof. Dr. Juan Gabriel Colonna pelo direcionamento, paciência, confiança e conselhos no desenvolvimento desta dissertação. Agradeço, na pessoa do Prof. Dr. Eduardo Luzeiro Feitosa, aos professores e ao corpo administrativo do PPGI pelo suporte e acompanhamento durante o mestrado.

Agradeço aos meus amigos e colegas do PPGI, Ada Cruz, Larissa Neves, Leandro Okimoto, Marcelo Chamy e Thais Gomes, pelo apoio e momentos de descontração.

Agradeço ao Instituto de Desenvolvimento Sustentável Mamirauá, na pessoa do Dr. Emiliano Esterci Ramalho, que gentilmente cedeu a base de dados Mamirauá utilizada neste trabalho.

As bases de dados de Caxiuanã e Central Suriname utilizadas nesta pesquisa foram fornecidas pelo Tropical Ecology Assessment and Monitoring (TEAM) Network, uma colaboração entre Conservation International, o Missouri Botanical Garden, o Smithsonian Institution e a Wildlife Conservation Society, e parcialmente financiado por estas instituições, pela Gordon and Betty Moore Foundation e outros doadores. Agradeço à Fernanda Santos e à Dr^{a}. Marcela Lima, responsáveis pelo projeto TEAM em Caxiuanã, que gentilmente permitiram a utilização da base Caxiuanã para o treinamento de modelos de identificação de animais desta pesquisa. Agradeço ao Conservation International Suriname que forneceu acesso à base Central Suriname.

Agradeço ao Instituto de Desenvolvimento Sustentável Mamirauá, através do Projeto Providence, à empresa Méliuz, através do projeto Bolsa Méliuz, e à CAPES pelo apoio financeiro para a execução desta pesquisa.

"Parecia um aparelho absurdamente complicado, e esse era um dos motivos pelos quais a capa plástica do dispositivo trazia a frase NÃO ENTRE EM PÂNICO em letras grandes e amigáveis."

(O Guia do Mochileiro das Galáxias)

Resumo

A utilização de armadilhas fotográficas é uma estratégia de monitoramento da vida selvagem que consiste na instalação de câmeras com sensores de movimento que, ao serem acionados, ativam a gravação de curtas sequências de imagens ou vídeos de animais, sem interferir em seu comportamento natural. Essas câmeras obtêm milhões de imagens, mas a extração de informação é tradicionalmente feita por humanos, tarefa que demanda tempo e é dispendiosa. Técnicas de aprendizado profundo são o estado da arte para extração de informações a partir de imagens e têm sido aplicadas em diversos trabalhos para a classificação de animais em imagens de armadilhas fotográficas. Como esses modelos têm alta capacidade de representação e podem facilmente memorizar toda a base de treinamento, deve-se evitar sobreposição de imagens muito semelhantes nas bases de treino e de teste, a fim de avaliar corretamente a capacidade de generalização dos modelos. Entretanto, a similaridade entre as imagens de armadilhas fotográficas obtidas em um mesmo local em curtos períodos de tempo tem recebido pouca atenção na literatura da área. O particionamento aleatório dos dados é a abordagem mais comum utilizada nos trabalhos que investigam a classificação de espécies em imagens de armadilhas fotográfica. Porém, esse tipo de abordagem pode gerar conjuntos de teste otimistas em relação às condições reais de utilização dos modelos, fato que pode implicar em uma avaliação superestimada dos modelos treinados e pode levar à tomada de decisões equivocadas. Considerando esse contexto, neste trabalho foi realizado um estudo sobre abordagens de particionamento de dados entre treino e teste em bases de classificação de espécies de animais em imagens de armadilhas fotográficas a fim de reduzir o viés otimista na construção de conjuntos de teste. Cenários reais de utilização foram simulados e avaliados para verificar se os conjuntos de teste conseguem evidenciar a capacidade de generalização dos modelos nessas condições. Como resultado, foi especificado um conjunto de recomendações para o particionamento dos dados para avaliação *out-of-sample* de modelos de acordo com o protocolo utilizado pelo projeto de armadilhas fotográficas.

Palavras-chave: Redes neurais profundas, armadilhas fotográficas, aprendizagem de máquina, particionamento dos dados, monitoramento da vida selvagem

Abstract

Camera traps are a strategy for wildlife monitoring, which consists on using cameras with motion sensors that, when triggered, start recording short sequences of images or videos of animals without disturbing their natural behavior. These cameras capture millions of images, but the information extraction is traditionally performed by humans, which is an expensive and time-consuming manual task. Deep learning techniques are the state of the art for extracting information from images and have been applied in several works to perform animal species classification in camera trap images. Since these models have high representation capacity and can easily memorize the entire training set, overlapping of very similar images in training and test sets should be avoided, in order to correctly evaluate the models generalization capacity. However, the possible high similarity between camera trap images obtained at the same place in short periods of time has not received a great deal of attention in the literature. The random data splitting is the the most widely used strategy in works dealing with animal species classification in camera trap images. Nevertheless, this strategy may generate optimistic test sets when compared to the actual conditions of use, which may result in an overestimated assessment of the trained model and may lead to wrong decisions. Therefore, we conduct in this work a study related to dataset splitting approaches for camera trap datasets, in order to reduce the optimistic bias of the test sets. Real usage scenarios were simulated and evaluated to verify whether or not the test sets are able to show the generalization capacity of the models under these conditions. As a result, a set of recommendations for dataset splitting on out-of-sample evaluation of models was specified according to the protocol used by the camera trap projects.

Keywords: Deep neural networks, camera traps, machine learning, dataset splitting, wildlife monitoring

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

cLBP *Cell Structured Local Binary Patterns*

EVOC *Ensemble Video Object Cut*

GPU *Graphics Processing Unit*

ILSVRC *ImageNet Large-Scale Visual Recognition Challenge*

ReLu *Rectified Linear Unit*

ResNet *Residual Networks*

ScSPM *Sparse Coding Spatial Pyramid Matching*

SGD *Stochastic Gradient Descent*

SIFT *Scale Invariant Feature Transform*

SVM *Support Vector Machine*

Sumário

1

Introdução

O monitoramento da vida selvagem é utilizado por pesquisadores para a obtenção de dados essenciais que ajudam, por exemplo, a compreender a ecologia das mais diversas espécies, a avaliar o impacto da ação humana e das mudanças climáticas no ecossistema, entre outras questões [Ahumada et al., 2013, He et al., 2016b, Kays et al., 2015]. Com isso, é possível estabelecer estratégias que visem à conservação de espécies, como a criação e gestão de unidades de conservação, bem como o uso sustentável dos recursos da natureza.

A utilização de armadilhas fotográficas para o monitoramento da vida selvagem se tornou popular nos últimos anos devido ao baixo custo, rápida instalação e fácil manutenção em comparação com outras abordagens [He et al., 2016b]. Essas armadilhas são câmeras com sensores de movimento que, ao serem acionados, ativam a gravação de curtas sequências de imagens ou vídeos de animais, sem interferir em seu comportamento natural. Enquanto essas câmeras obtêm milhões de imagens, a extração de informação dessas imagens é tradicionalmente feita por seres humanos (especialistas ou uma comunidade de voluntários, por exemplo), tarefa que demanda tempo e é dispendiosa [Norouzzadeh et al., 2018]. Assim, a automatização dessa extração de informação pode acelerar o trabalho dos profissionais, permitindo-lhes focar nas suas missões científicas.

1.1 Motivação

A principal informação extraída a partir de imagens de armadilhas fotográficas é a espécie do animal fotografado. Além disso, também é comum realizar a contagem da quantidade de indivíduos presentes, bem como descrever seu comportamento na cena [Kays et al., 2009, Swanson et al., 2015b]. Em alguns projetos, quando possível, são

catalogados também o sexo e a idade dos indivíduos. Também podem ser calculadas velocidade e direção de deslocamento a partir da movimentação observada nas imagens de um mesmo evento de captura – que corresponde ao conjunto de imagens obtidas a cada vez que o sensor de movimento é acionado [He et al., 2016b]. Dependendo da espécie, é possível ainda identificar um indivíduo específico por meio de marcas únicas na pele, como no caso das onças pintadas.

Essas informações são utilizadas pela comunidade científica na análise de diversas questões ecológicas, como a comprovação da presença de espécies raras em um local, estimação da densidade populacional e ocupação territorial, determinação de padrões de comportamento, dentre outras [Burton et al., 2015]. Quanto mais dados de armadilhas fotográficas são acumulados em vários pontos de captura e ao longo de vários anos, mais relevantes eles se tornam. Eles podem mostrar, por exemplo, as complexas relações entre presas e predadores, registrando a demografia de suas populações ao longo dos anos [Kays et al., 2009].

O incremento na quantidade de pontos de captura e tempo de monitoramento, no entanto, implica diretamente no aumento da quantidade de imagens a serem analisadas, o que não é escalável se essa análise for realizada manualmente. Por exemplo, o projeto Snapshot Serengeti instalou armadilhas fotográficas em 225 pontos de captura no Parque Nacional do Serengeti, Tanzânia, coletando entre junho de 2010 e maio de 2013 cerca de 3,2 milhões de imagens. A extração de informações dessa enorme quantidade de imagens só foi possível graças à colaboração de uma comunidade online de mais de 28 mil voluntários [Swanson et al., 2015b].

Entretanto, conseguir uma comunidade de voluntários de tamanho expressivo como a do Snapshot Serengeti é inviável para projetos de menor porte ou com menos exposição publicitária. Assim sendo, a automatização da análise de imagens de armadilhas fotográficas tem grande potencial para acelerar estudos ecológicos, podendo ser essencial para a viabilidade de determinados projetos. Mesmo que a análise automática não seja 100% precisa, ela pode tornar a extração de informações consideravelmente mais eficiente, por exemplo, eliminando imagens sem animais e apresentando sugestões das espécies mais prováveis para validação manual [He et al., 2016b].

Nos últimos anos, abordagens que usam arquiteturas de redes neurais profundas têm sido o estado da arte para o problema de classificação de imagens, atingindo, ou até mesmo ultrapassando, as taxas de acerto humanas em determinadas bases de dados [He et al., 2016a, Szegedy et al., 2017, 2016]. Tendo isso em vista, alguns trabalhos têm aplicado redes neurais profundas para a classificação de imagens de armadilhas fotográficas, obtendo resultados promissores [Norouzzadeh et al., 2018, Tabak et al., 2018, Villa et al., 2016, 2017, Willi et al., 2018].

1.2 Redes neurais profundas para classificação de imagens de armadilhas fotográficas

A automatização da extração de informações de imagens de armadilhas fotográficas tem se concentrado em duas tarefas: classificar se a imagem contém um animal ou somente plano de fundo, e, caso haja animal na cena, identificar a espécie. Há duas abordagens principais no tratamento dessas tarefas. A primeira abordagem consiste em utilizar um modelo onde as classes são as espécies de interesse e uma classe de plano de fundo [Tabak et al., 2018]. A segunda é utilizar dois estágios: o modelo do primeiro estágio é treinado para classificar se a imagem contém um animal ou somente plano de fundo, enquanto o segundo estágio é treinado somente para classificar a espécie do animal [Norouzzadeh et al., 2018, Willi et al., 2018]. O presente trabalho tem foco nos modelos para o segundo estágio.

Geralmente, para aumentar as chances de se capturar uma imagem que permita identificar o animal na cena, obtém-se mais de uma imagem por evento de captura, com um intervalo de um segundo entre elas. Agregando-se a isso o fato que imagens obtidas em um mesmo ponto de captura costumam ser semelhantes por compartilharem o mesmo plano de fundo, as imagens de bases de armadilhas têm um alto grau de similaridade. A Figura 1.1 mostra um exemplo dessa similaridade das imagens obtidas no mesmo evento de captura.

Figura 1.1: Exemplo de imagens similares obtidas em um mesmo evento de captura.

Fonte: Base de Imagens Snapshot Serengeti [Swanson et al., 2015a].

No entanto, como modelos de aprendizado profundo têm alta capacidade de representação e podem facilmente memorizar bases inteiras, deve-se evitar sobreposição de imagens muito semelhantes nas partições de dados de treino e de teste. Contudo, grande parte dos trabalhos ignoram essa similaridade das imagens e realizam o particionamento de maneira totalmente aleatória [Chen et al., 2014, Tabak et al., 2018, Villa et al., 2016, 2017, Yu et al., 2013]. Há, porém, trabalhos que levam em consideração

essa questão. Por exemplo, nos trabalhos de Norouzzadeh et al. [2018] e de Willi et al. [2018], o particionamento em treino e teste é realizado colocando-se imagens de um mesmo evento de captura na mesma partição. Entretanto, essa estratégia pode não ser suficiente, uma vez que vários eventos de captura de uma mesma espécie no mesmo local podem conter imagens muito similares entre si.

O fato das câmeras serem fixas limita o escopo de treinamento e utilização dos modelos. Entretanto, com o passar do tempo, novas situações que fogem da amostragem da base (*out-of-sample*) podem ocorrer, como a mudança natural da vegetação, a mudança do ângulo de visão da câmera durante uma operação de manutenção do equipamento, a captura de imagens de espécies não observadas anteriormente em determinados locais, entre outras. O simples particionamento aleatório ou agrupamento por evento pode não ser capaz de evidenciar se os modelos treinados são robustos o suficiente para lidar com essas situações.

Nesse caso, a utilização de um conjunto de teste otimista pode implicar em uma avaliação superestimada do modelo treinado e pode levar à tomada de decisões equivocadas. A otimização dos parâmetros de treinamento de acordo com o desempenho em uma base de validação pode ser ineficaz, devido à base não representar condições realistas de utilização. Willi et al. [2018] fazem um estudo para a escolha de um limiar de confiança para que as predições do modelo sejam levadas em consideração. Como a escolha desse limiar é realizada conforme uma relação de compromisso entre a precisão e a revocação do modelo, o nível escolhido pode ser muito alto para condições reais, implicando em uma revocação abaixo do previsto.

Portanto, é necessário desenvolver um estudo para a construção de bases de teste que ofereçam condições de avaliação mais próximas às situações reais de utilização dos modelos, dadas as especificidades das bases de imagens de armadilhas fotográficas.

Para o treinamento de redes neurais profundas, é necessária uma grande quantidade de instâncias, no entanto, a quantidade de dados rotulados disponíveis em projetos de armadilhas fotográficas de pequena escala é bastante limitada. Em cenários desse tipo, as arquiteturas profundas tendem a fazer *overfitting* nos dados de treinamento [Donahue et al., 2014]. Uma alternativa para contornar essa situação consiste em utilizar transferência de aprendizado, onde o conhecimento aprendido por um modelo em determinado problema é utilizado para melhorar o desempenho em um problema similar [Pan & Yang, 2010].

Frequentemente, utiliza-se modelos pré-treinados na base ImageNet [2015] para transferir aprendizado para os mais diversos domínios de problemas de visão computacional. Essa estratégia foi adotada por Villa et al. [2017] para classificar animais da base Snapshot Serengeti. Por outro lado, partindo do princípio de que é mais fácil

transferir aprendizado entre problemas do mesmo domínio, Willi et al. [2018] utilizam modelos pré-treinados na Snapshot Serengeti para transferir aprendizado para outras bases de armadilhas fotográficas de menor porte.

Entretanto, a baixa variabilidade das imagens de armadilhas fotográficas pode prejudicar o aprendizado de características com boa capacidade de representação. Somando-se à hipótese de que o conjunto de teste possui viés otimista, a real capacidade de transferir aprendizado de modelos treinados em bases de armadilhas fotográficas pode ser superestimada. Em contrapartida, modelos treinados na ImageNet normalmente aprendem características com alta capacidade de representação que funcionam como descritores de imagem para as mais diversas tarefas de reconhecimento visual [Donahue et al., 2014, Oquab et al., 2014]. Portanto, é necessário um estudo que verifique quais modelos-base oferecem melhores condições de transferência de aprendizado para classificação de imagens de armadilhas fotográficas.

A automatização da extração de informações das imagens obtidas com armadilhas fotográficas pode incentivar a expansão de projetos de monitoramento desse tipo, com a adição de novos pontos de captura. Entretanto, o desempenho dos modelos pode cair ao serem utilizados para classificar imagens das novas localizações. Willi et al. [2018] sugerem que os modelos podem aprender características específicas ou vieses das espécies em relação a locais específicos. Assim, ao ser utilizado em imagens de novos pontos de captura, um modelo perderia acurácia, entretanto, essa possibilidade não foi avaliada no trabalho de Willi et al. [2018]. Portanto, diante da possibilidade de utilização dos modelos para classificar imagens de novos pontos de captura, faz-se necessário um estudo mais aprofundado da capacidade de generalização dos modelos, dado o viés em relação aos locais utilizados no treinamento.

1.3 Hipótese de pesquisa

As abordagens de particionamento de dados em treinamento e teste de bases de armadilhas fotográficas realizadas por meio da seleção aleatória de imagens obtidas em eventos de captura, sem levar em conta os contextos temporal e espacial, não garante de forma suficiente a independência das amostras de treino e teste, o que leva ao *overfitting*, uma baixa capacidade de representação e a uma dificuldade em tarefas como a transferência de aprendizado.

Com base nessa hipótese, a próxima seção apresenta os objetivos deste trabalho.

1.4 Objetivos

A pesquisa desta dissertação tem como objetivo geral propor e validar abordagens de particionamento de bases de imagens de armadilhas fotográficas para reduzir o viés otimista do conjunto de teste visando oferecer condições de avaliação mais próximas às situações reais de utilização dos modelos.

Para atingir este objetivo geral, pretende-se alcançar os seguintes objetivos específicos:

- Identificar situações *out-of-sample* com as quais um modelo pode ter que lidar em condições reais de utilização e que possam ser subestimadas pelo viés otimista do conjunto de teste.
- Demonstrar experimentalmente a capacidade de generalização dos modelos para classificar animais em imagens de pontos de captura não incluídos no treinamento.
- Especificar um conjunto de recomendações para avaliação *out-of-sample* de modelos de acordo com o protocolo utilizado pelo projeto de armadilhas fotográficas.
- Verificar quais modelos-base oferecem melhores condições de transferência de aprendizado para classificação de animais em imagens de armadilhas fotográficas.

1.5 Organização da dissertação

Esta dissertação está organizada da seguinte forma: o Capítulo 2 apresenta os fundamentos teóricos necessários para entendimento dos métodos adotados; no Capítulo 3 é feita uma síntese dos trabalhos relacionados; as abordagens propostas para avaliação *out-of-sample* são apresentadas no Capítulo 4, onde é realizado um estudo de caso na base Snapshot Serengeti; no Capítulo 5 é realizado um estudo sobre transferência de aprendizado entre bases de armadilhas fotográficas utilizando-se as recomendações da abordagem de particionamento proposta; por fim, o Capítulo 6 apresenta as conclusões do trabalho, contribuições, limitações e direções futuras.

2

Fundamentos Teóricos

Neste capítulo são apresentados os fundamentos sobre redes neurais profundas e transferência de aprendizado. Esses conceitos serão fundamentais para a compreensão do capítulo de trabalhos relacionados, bem como das abordagens de avaliação propostas para o problema classificação de animais em imagens de armadilhas fotográficas.

Nas seções a seguir é feita uma descrição do princípio de funcionamento das redes neurais convolutivas, principal tipo de rede neural aplicado para problemas de visão computacional (Seção 2.1), e as estratégias que possibilitam o treinamento de arquiteturas mais profundas (Seções 2.2 e 2.3). Também são apresentadas as principais arquiteturas de redes convolutivas utilizadas na literatura, indicando quais os avanços obtidos por cada uma (Seção 2.4).

Em seguida, é feita uma descrição da técnica de transferência de aprendizado utilizando-se redes neurais convolutivas (Seção 2.5). Esse tipo de técnica é fundamental para treinar modelos profundos para bases com quantidade não muito expressiva de amostras rotuladas, como é o caso de boa parte das bases de imagens de armadilhas fotográficas.

Por fim, a Seção 2.6 faz uma descrição do funcionamento de algoritmos genéticos que foram utilizados para o particionamento em treino e teste das bases, e a Seção 2.7 descreve as métricas de avaliação utilizadas nesta dissertação.

2.1 Redes Convolutivas

Redes neurais artificiais são modelos computacionais inspirados na estrutura do sistema nervoso de animais que adquire conhecimento por meio da experiência. Uma rede neural *feedfoward* é um tipo de rede cujo objetivo é aproximar alguma função f^* e o processamento da informação flui num único sentido, da camada de entrada para a

camada de saída, não possuindo conexões de retroalimentação da saída das camadas posteriores para as anteriores [Goodfellow et al., 2016].

Redes neurais convolutivas são um subtipo especial de rede neural *feedfoward* especializado no processamento de dados organizados em topologia similar a uma grade [Goodfellow et al., 2016]. O exemplo clássico desse tipo de dado são as imagens, as quais são organizadas em topologia 2D. Outros exemplos que podem ser citados são os vídeos (topologia 3D com imagens ao longo do tempo) e sequências de sinais de áudio (topologia 1D) [LeCun et al., 2015].

A arquitetura de uma rede neural convolutiva é estruturada como uma série de camadas empilhadas [LeCun et al., 2015], conforme exemplifica a Figura 2.1. Os principais tipos de camadas utilizados são as camadas convolutivas, as camadas de *pooling* e as camadas completamente conectadas.

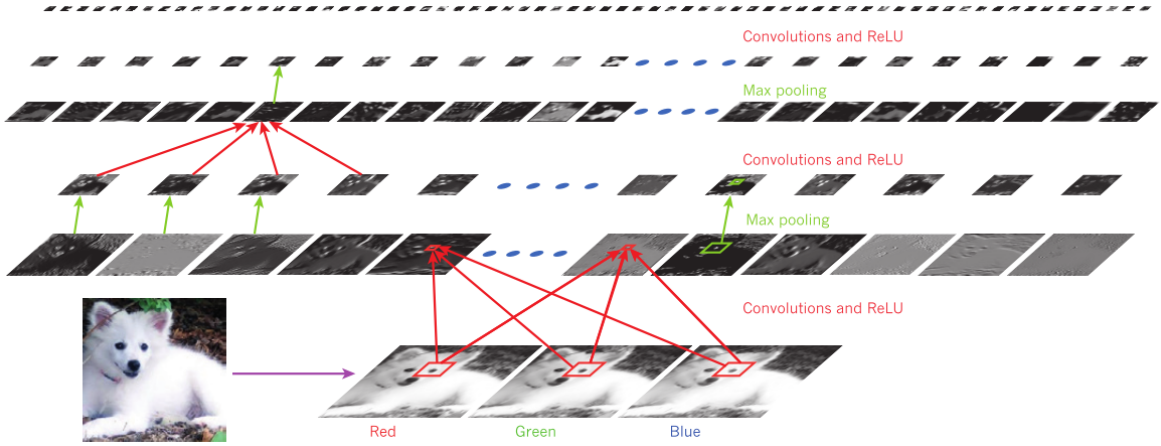


Figura 2.1: Esquema de uma arquitetura de rede convolutiva.

Fonte: LeCun et al. [2015].

As camadas convolutivas tomam vantagem de dois conceitos principais: campos receptivos locais e pesos compartilhados. Campos receptivos locais são pequenas regiões de uma imagem que fornecem informações localizadas. Cada unidade de uma camada convolutiva está conectada somente a um conjunto de campos receptivos locais, em vez de todas as unidades da camada anterior, como ocorre nas camadas completamente conectadas. Além disso, essas unidades são organizadas em mapas de filtros, onde as unidades de cada mapa compartilham os mesmos pesos. Essa organização permite que a rede aprenda características locais distintas ao longo dos campos receptivos e, ao mesmo tempo, reduz drasticamente a quantidade de pesos necessários por camada [LeCun et al., 2015].

Camadas de *pooling* funcionam fazendo sub-amostragem da representação recebida, juntando características semanticamente similares em uma só. Uma unidade típica de uma camada de *pooling* calcula o valor máximo de um conjunto local de unidades de um determinado mapa de filtros. Outra operação comum nesse tipo de camada é o cálculo da média do conjunto local. Esse tipo de operação reduz significativamente a quantidade de parâmetros passados adiante e, consequentemente, melhora a eficiência computacional de toda a arquitetura [Goodfellow et al., 2016]. No entanto, em arquiteturas mais recentes as camadas de *pooling* têm sido substituídas por camadas convolutivas com *stride* maior, isto é, utiliza-se um passo mais largo durante a operação de convolução do filtro sobre a entrada da camada [He et al., 2016a].

Outra característica das redes convolutivas modernas é a utilização de conexões com saltos, onde a saída de uma camada pode ser utilizada como entrada tanto para a camada imediatamente seguinte quanto para outras posteriores na arquitetura [He et al., 2016a].

Conforme mais camadas são empilhadas, a rede aprende representações cada vez mais abstratas da imagem. Geralmente, adiciona-se camadas completamente conectadas ao final da rede que utilizam essas representações para aprender a tarefa para a qual a rede está sendo treinada como, por exemplo, o reconhecimento de objetos em imagens [Goodfellow et al., 2016].

Com o surgimento de novas técnicas de treinamento de redes neurais, como a função de ativação ReLu, o método de regularização *dropout* e o *batch normalization*, passou a ser possível treinar redes neurais mais profundas, isto é, com mais camadas empilhadas. Redes convolutivas profundas que possuem, por exemplo, uma profundidade de 5 a 20 camadas empilhadas, podem implementar funções muito complexas que conseguem ao mesmo tempo ser sensíveis a detalhes minuciosos – distinguindo samoiedos de lobos brancos – e insensíveis a grandes variações irrelevantes tais como plano de fundo, iluminação e objetos nas redondezas [LeCun et al., 2015].

2.2 Otimizadores para Treinamento de Redes Neurais Profundas

O treinamento de redes neurais profundas é realizado através de algoritmos otimizadores que buscam os parâmetros $\boldsymbol{\theta}$ do modelo de tal forma que minimizem uma determinada função de custo $J(\boldsymbol{\theta})$ que é avaliada no conjunto de dados de treinamento. No entanto, diferente dos problemas clássicos de otimização, cujo objetivo é minimizar a função J em si, a otimização em aprendizagem de máquina busca minimizar J

na esperança de melhorar o desempenho D do modelo em um conjunto de teste com amostras não previamente apresentadas ao modelo [Goodfellow et al., 2016].

A grande maioria dos otimizadores utilizados em aprendizado profundo são baseados no método do gradiente descendente. Nesse método, os parâmetros $\boldsymbol{\theta}$ do modelo são atualizados em pequenos passos, movendo-os na direção oposta do sinal do gradiente da função de custo J, a qual é minimizada iterativamente [Robbins & Monro, 1951].

Otimizadores que utilizam o conjunto de treinamento inteiro a cada passo do cálculo incremental são comumente chamados de métodos determinísticos. Já os otimizadores que utilizam uma única amostra por passo são normalmente chamados de métodos estocásticos. A grande maioria dos algoritmos utilizados em aprendizado profundo utiliza em cada iteração um lote com mais de uma amostra mas com menos que o conjunto total de treinamento. Esses algoritmos são comumente chamados de métodos estocásticos em mini-lotes, mas recentemente passaram a ser simplesmente chamados de métodos estocásticos [Goodfellow et al., 2016].

O algoritmo de otimização SGD (do inglês *Stochastic Gradient Descent*) é um dos algoritmos mais utilizados em aprendizado profundo, atualizando os parâmetros do modelo de acordo com o método do gradiente descendente. Nesse algoritmo, o hiperparâmetro mais importante é a taxa de aprendizado, isto é, o tamanho do passo utilizado na atualização dos parâmetros. No entanto, vários problemas podem ocorrer durante a otimização, tais como o gradiente ficar preso em mínimos locais e pontos de sela. Várias técnicas foram desenvolvidas para combater esses problemas, como a utilização de momento no gradiente e algoritmos que adaptam a taxa de aprendizado no decorrer do treinamento de acordo com determinados critérios observados [Goodfellow et al., 2016]. Entre esses algoritmos podem ser citados o AdaGrad [Duchi et al., 2011], o RMSProp [Hinton et al., 2012] e o Adam [Kingma & Ba, 2014].

2.3 Regularizadores

Uma das questões centrais em aprendizagem de máquina é conseguir fazer com que os algoritmos tenham bom desempenho não somente na base de treino mas também em novas amostras. Há várias estratégias em aprendizagem de máquina que têm por objetivo reduzir o erro de teste, as quais são coletivamente chamadas de regularizadores [Goodfellow et al., 2016]. Entre as estratégias que têm sido fundamentais para o treinamento de redes profundas podem ser citadas *dropout* e o aumento de dados. A parada antecipada do treinamento também pode ser considerada uma técnica de regu-

larização, sendo utilizada para evitar *overfitting* no conjunto de treino. O método de *batch normalization* utilizado para melhorar a estabilidade de redes neurais profundas também atua como regularizador [Szegedy et al., 2016].

2.3.1 *Dropout*

Dropout é uma técnica que ajuda no combate ao *overfitting*, que consiste em remover temporariamente unidades de uma rede neural e suas conexões de saída. A cada apresentação de uma entrada ou conjunto de entradas, no caso de treinamento com mini-lotes, cada unidade da rede é aleatoriamente omitida com uma probabilidade fixa p independentemente das demais unidades. O valor de p pode ser escolhido a partir de uma base de validação ou, como na maioria dos casos, ser fixado em 0,5. Esse tipo de abordagem previne que co-adaptações complexas ocorram ao fazer com que não haja confiança quanto à presença de qualquer unidade em particular [Srivastava et al., 2014].

O *dropout* também pode ser visto como uma técnica para treinar eficientemente uma quantidade exponencial de redes neurais diferentes. Especificamente, a cada passo do treinamento, uma sub-rede é amostrada e treinada. Ao final do treinamento, obtém-se um modelo que é o conjunto de todas as sub-redes que podem ser formadas a partir da rede base com extenso compartilhamento de pesos. Conforme demonstrado por Srivastava et al. [2014], essa técnica melhora significativamente o desempenho de redes neurais nas mais diversas tarefas de aprendizado supervisionado.

2.3.2 Aumento de dados

A melhor maneira de fazer com que um modelo de aprendizado de máquina generalize melhor, reduzindo o *overfitting*, é realizar o treinamento com mais dados. Quando não é possível obter mais dados ou for muito dispendioso, uma alternativa consiste no aumento artificial da base de dados através de transformações que preservem os rótulos das amostras. Para o domínio de reconhecimento e classificação de objetos em imagens, as técnicas de aumento de dados têm sido particularmente efetivas [Goodfellow et al., 2016]. Imagens em geral permitem uma enorme diversidade de fatores de variações que podem facilmente ser simuladas. Operações como translação, rotação, mudança de escala e espelhamento têm se mostrado efetivas para aumento artificial de bases de dados de imagens [Goodfellow et al., 2016].

2.3.3 Parada antecipada

A parada antecipada é das formas mais simples de regularização, permitindo finalizar o treinamento quando detecta-se que o modelo está fazendo *overfitting* no conjunto de treinamento. Esse *overfitting* pode ser constatado quando o desempenho do modelo no conjunto de dados de treinamento continua a melhorar enquanto que o desempenho no conjunto de validação passa a se deteriorar. Assim, essa estratégia encerra o treinamento antecipadamente quando o desempenho no conjunto de validação não melhora depois de algum tempo. Nesse caso, escolhe-se o modelo que obteve o melhor desempenho no conjunto de validação, na esperança de obter-se o melhor erro de generalização no conjunto de teste [Goodfellow et al., 2016].

2.3.4 *Batch normalization*

O método do gradiente descendente utilizado para treinar redes neurais indica como atualizar os parâmetros de uma camada, levando-se em consideração que as demais camadas permaneçam inalteradas. Na prática, todas as camadas são atualizadas simultaneamente. Isso gera efeitos de segunda e terceira ordens que tornam o treinamento mais difícil, pois cada camada depende muito de como as demais camadas variam [Goodfellow et al., 2016].

O método de normalização em lotes, em inglês *batch normalization*, foi desenvolvido para combater esse problema, garantindo estabilização dos parâmetros e reduzindo drasticamente o tempo necessário para o treinamento. Esse método pode ser aplicado a qualquer camada de uma rede e funciona normalizando as ativações das unidades pela média e desvio padrão em cada lote [Ioffe & Szegedy, 2015]. Com isso, cada camada adquire uma certa independência em relação às demais camadas durante o treinamento.

Como o cálculo dos desvios e médias é feito a cada lote, um ruído é introduzido na rede. Esse ruído tem efeito similar ao *dropout*, fazendo com que as unidades tenham que aprender a serem robustas a muitas variações nas suas entradas. Dessa forma, o *batch normalization* atua também como um leve regularizador, podendo dispensar o uso de *dropout*, dependendo do caso [Szegedy et al., 2016].

2.4 Arquiteturas de Redes Profundas

Na literatura há diversas arquiteturas de redes convolutivas profundas projetadas para os problemas de visão computacional. Nesta seção, são descritas as arquiteturas Alex-

Net, VGGNet, GoogLeNet, ResNet e InceptionV3, assim como as inovações mais relevantes introduzidas por cada uma delas.

2.4.1 AlexNet

A arquitetura proposta por Krizhevsky et al. [2012] foi responsável por popularizar o uso de redes convolutivas para os problemas de visão computacional ao ganhar o desafio *ImageNet Large-Scale Visual Recognition Challenge (ILSVRC)* em 2012. Essa arquitetura, hoje conhecida como AlexNet, foi responsável por reduzir em quase 10% a taxa de erro top-5 em relação ao segundo colocado, sendo um marco para a linha de pesquisa em aprendizagem profunda.

Conforme ilustra a Figura 2.2, essa arquitetura possui 8 camadas com parâmetros aprendidos durante o treinamento, sendo 5 camadas convolutivas seguidas de 3 camadas completamente conectadas, totalizando cerca de 60 milhões de parâmetros. A primeira camada convolutiva da rede recebe como entrada uma imagem de 224 x 224 x 3 (imagem de 224 pixels de altura por 224 pixels de largura, com 3 canais de cor RGB). Após determinadas camadas convolutivas são adicionadas camadas de *pooling*, fazendo redução de dimensionalidade. Por fim, a saída da última camada completamente conectada é usada como entrada para a função softmax que produz a distribuição de probabilidade para as 1000 classes do desafio ILSVRC.

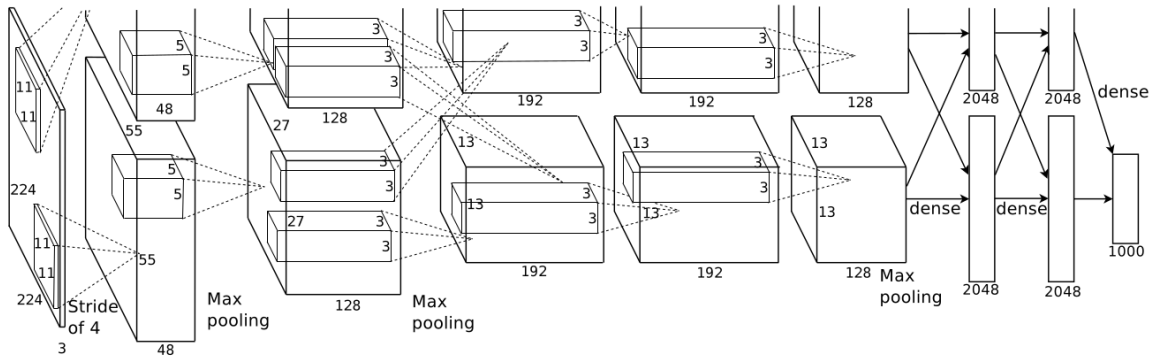


Figura 2.2: Ilustração da arquitetura da AlexNet.

Fonte: Krizhevsky et al. [2012].

O sucesso dessa arquitetura se deu por uma combinação de várias técnicas até então pouco usuais nas arquiteturas de redes neurais. Foi utilizada como função de ativação dos neurônios a função ReLu, fazendo com que o treinamento fosse muitas vezes mais rápido que as redes equivalentes que usam a função Tangente Hiperbólica. Outro fator que teve grande importância foi a utilização de placas de processamento

gráfico (GPUs, do inglês *Graphics Processing Unit*) para realizar o treinamento, permitindo uma computação massiva em paralelo. Para combater o *overfitting*, os autores aplicaram as técnicas de aumento de dados e *dropout*.

2.4.2 VGGNet

A arquitetura VGG proposta por Simonyan & Zisserman [2014] melhorou significativamente a taxa de erro top-5 em relação a AlexNet, atingindo 6,8%. A grande contribuição dessa arquitetura foi aumentar a profundidade da rede com filtros convolutivos muito pequenos (3 x 3), atingindo 16 a 19 camadas de pesos, o que na época foi considerado muito profundo. Essa arquitetura possui o mesmo tamanho da entrada que a AlexNet (224 x 244 x 3), bem como a mesma quantidade de neurônios nas camadas completamente conectadas.

A grande diferença desta arquitetura consiste em substituir os grandes filtros receptivos das primeiras camadas (11 x 11 e 5 x 5 para a primeira e segunda camadas da AlexNet, respectivamente) por filtros muito pequenos de tamanho 3 x 3 em todas as camadas. O empilhamento de duas camadas de filtros 3 x 3 sem *pooling* entre elas produz um campo receptivo equivalente a um filtro 5 x 5; e três camadas de filtros 3 x 3 produzem um campo equivalente a um filtro 7 x 7. Essa configuração traz duas vantagens: primeiro, incorpora três camadas retificadoras não-lineares em vez de uma, o que torna a função de decisão mais discriminativa; uma segunda vantagem é que essa configuração reduz drasticamente a quantidade de parâmetros a serem aprendidos, tendo cerca de 144 milhões de parâmetros na configuração mais profunda com 19 camadas.

2.4.3 GoogLeNet

GoogLeNet é uma versão particular da arquitetura Inception que venceu o desafio ILSVRC em 2014, atingindo 6,67% de taxa de erro top-5, superando a VGGNet [Szegedy et al., 2015]. Essa arquitetura foi construída com o objetivo de otimizar o uso dos recursos computacionais dentro da rede. Através de um *design* cuidadosamente elaborado, Szegedy et al. [2015] conseguiram aumentar a profundidade da rede e a quantidade de unidades por camada, utilizando doze vezes menos parâmetros que a AlexNet.

Além das camadas convolutivas e de *pooling*, a GoogLeNet utiliza um novo módulo chamado Inception baseado na ideia de redes dentro de redes proposta por Lin et al. [2013]. Nesse módulo, em vez de cada camada executar ou uma operação de *pooling* ou de convolução de um tamanho específico, várias dessas operações são executadas

em paralelo, conforme ilustra a Figura 2.3. Além disso, esse módulo inclui conceitos de conexão esparsa através da adição de filtros convolutivos 1 x 1 a fim de fazer redução de dimensionalidade.

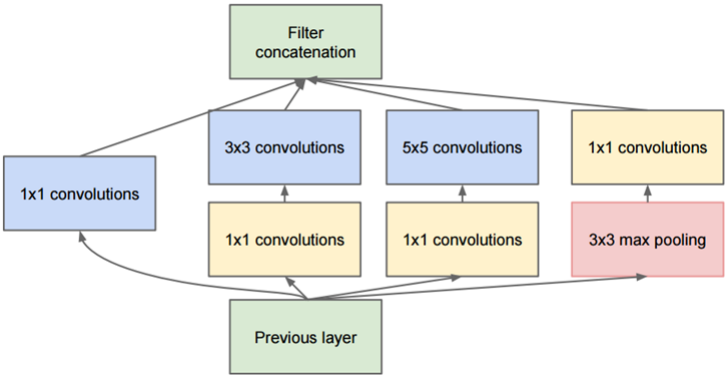


Figura 2.3: Módulo Inception.

Fonte: Szegedy et al. [2015].

A arquitetura da GoogLeNet é composta, portanto, de vários desses módulos Inception empilhados, conforme ilustra a Figura 2.4, totalizando 22 camadas de profundidade contando somente aquelas com parâmetros e 27 se levar as de *pooling* em consideração. O número total de camadas ou blocos independentes é aproximadamente 100, variando de acordo como as camadas são contadas.

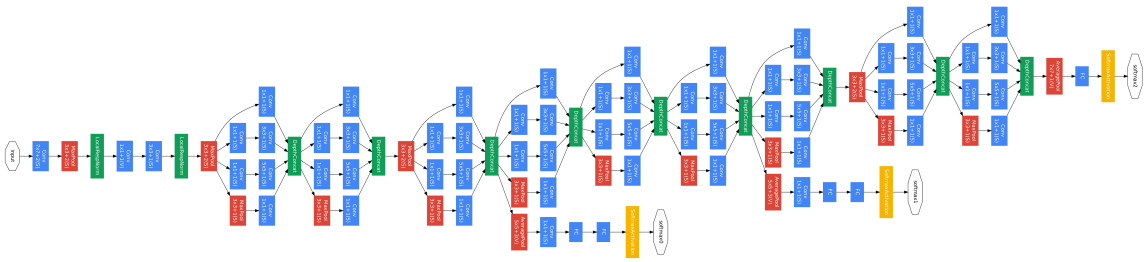


Figura 2.4: Arquitetura da GoogLeNet.

Fonte: Szegedy et al. [2015].

A fim de combater o problema do desaparecimento do gradiente que ocorre em decorrência da profundidade da rede, a GoogLeNet utiliza dois classificadores auxiliares na forma de pequenas redes convolutivas sobre os módulos Inception 4a e 4d. O erro desses classificadores é adicionado ponderadamente ao erro total da rede durante a etapa de treinamento. Na etapa de inferência da rede, esses classificadores auxiliares são completamente removidos.

2.4.4 ResNet

As arquiteturas descritas anteriormente evidenciam que a profundidade da rede tem importância crucial na melhora da acurácia do modelo. No entanto, quanto mais profunda for uma rede, mais difícil será treiná-la, devido ao problema de desaparecimento do gradiente, o qual é tratado de várias maneiras pelas arquiteturas anteriores [He et al., 2016a]. No entanto, de acordo com He et al. [2016a], há também um problema de degradação: conforme a profundidade da rede aumenta, a acurácia satura e então degrada rapidamente. No entanto, isso não ocorre devido ao *overfitting* e sim porque o erro de treinamento aumenta.

Para tratar esse problema de degradação, He et al. [2016a] propuseram as redes residuais profundas (ResNets) que adicionam ligações diretas entre as camadas. A Figura 2.5 apresenta uma ResNet com 34 camadas. Esse tipo de rede permitiu a construção de arquiteturas muito profundas com centenas de camadas. Uma versão com 152 camadas (ResNet-152), que é oito vezes mais profunda que a VGGNet, venceu o desafio ILSVRC em 2015, atingindo uma taxa de erro top-5 de 3,57%.

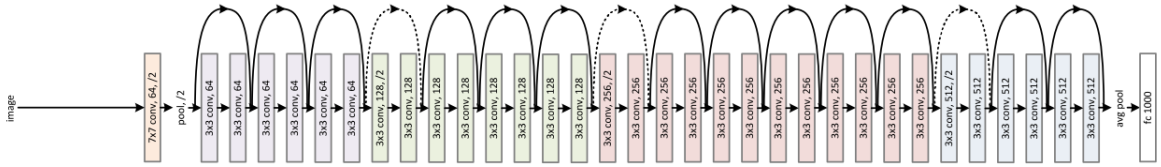


Figura 2.5: Arquitetura de uma ResNet com 34 camadas.

Fonte: He et al. [2016a].

A ideia principal de um bloco residual, representado na Figura 2.6, consiste em somar à saída de um conjunto de camadas empilhadas a entrada da primeira dessas camadas. Esse tipo de construção da rede permite que camadas mais profundas da rede recebam dados diretamente das camadas mais rasas. Desta forma, a rede pode aprender o quão profunda ela deve ser para aprender determinadas funções.

2.4.5 InceptionV3

A fim de escalar melhor as redes convolutivas, Szegedy et al. [2016] propuseram uma série de princípios a serem seguidos no *design* de arquiteturas profundas e fizeram estudos no contexto das redes Inception. Como resultado, foi proposta a arquitetura InceptionV3 que obteve, através de um conjunto de quatro modelos, uma taxa de erro top-5 de 3,58% na base de imagens do desafio ILSVRC de 2012. Apesar dessa

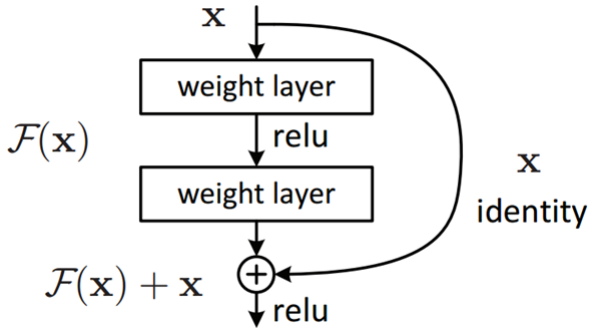


Figura 2.6: Bloco residual.

Fonte: He et al. [2016a].

arquitetura ter profundidade de 42 camadas, o custo computacional é apenas 2,5 vezes maior que o da GoogLeNet e continua muito mais eficiente que a VGGNet.

Um dos princípios adotados nesta arquitetura foi a fatoração de convoluções em convoluções de tamanho menor, similar à adotada pela VGGNet: convoluções de 7 x 7 foram fatoradas em três convoluções de 3 x 3 e convoluções de 5 x 5 em duas convoluções de 3 x 3. Também foi introduzida a fatoração espacial em filtros assimétricos, onde uma convolução n x n pode ser substituída por uma convolução 1 x n seguida por outra n x 1. Essa fatoração espacial, no entanto, não funciona muito bem para as primeiras camadas, mas apresenta bons resultados nas intermediárias [Szegedy et al., 2016].

Analisando os classificadores auxiliares presentes nas versões anteriores das arquiteturas Inception, Szegedy et al. [2016] chegaram à conclusão que essas ramificações da rede atuam como regularizadores e que a remoção do classificador auxiliar nas camadas mais rasas não afeta a qualidade final do modelo.

2.5 Transferência de aprendizado

Para que redes neurais profundas consigam generalizar é necessário treiná-las com bases de dados muito grandes. No entanto, em aplicações reais, como a identificação de animais em imagens de projetos de amardilhas fotográficas, nem sempre há bases rotuladas suficientemente grandes para treinar as arquiteturas modernas do zero, sem gerar *overfitting*. Nesses casos, uma alternativa consiste em utilizar transferência de aprendizado, onde os dados usados para realizar o treinamento não seguem a mesma distribuição da base de teste ou têm um espaço de características diferente [Pan & Yang, 2010].

A seguir será feita uma definição formal de transferência de aprendizado e na subseção seguinte serão abordadas as principais maneiras de transferência de aprendizado

utilizando as arquiteturas de redes neurais profundas.

2.5.1 Definição formal de transferência de aprendizado

Inicialmente, é necessário formalizar alguns conceitos para então definir transferência de aprendizado. As definições a seguir são baseadas no trabalho de Pan & Yang [2010].

Definição 1 *(Domínio). Um domínio* $D = \{\chi, P(X)\}$ *constitui-se de dois componentes: um espaço de características* χ *e uma distribuição de probabilidade marginal* $P(X)$*, onde* $X = \{x_1, ..., x_n\} \in \chi$*.*

Definição 2 *(Tarefa). Uma tarefa* $T = \{Y, f(\cdot)\}$ *constitui-se de dois componentes: um espaço de rótulos* $Y = \{y_1, ..., y_m\}$ *e uma função de predição objetiva* $f(\cdot)$ *que não é conhecida, mas que pode ser aprendida a partir dos dados de treinamento que se constituem de pares* $\{x_i, y_i\}$*, onde* $x_i \in X$ *e* $y_i \in Y$*. Após aprendida, a função* $f(\cdot)$ *pode ser usada para predizer o rótulo* $f(x)$ *de uma nova instância* x*.*

Seja o domínio de origem dos dados denotado por $D_s = \{(x_{s_1}, y_{s_1}), ..., (x_{s_n}, y_{s_n})\}$, onde $x_{s_i} \in \chi_s$ são os dados da instância e $y_{s_i} \in Y_s$ é o rótulo correspondente. De maneira similar, denota-se os dados do domínio de destino como $D_t = \{(x_{t_1}, y_{t_1}), ..., (x_{t_n}, y_{t_n})\}$, onde $x_{t_i} \in \chi_t$ são os dados da instância e $y_{t_i} \in Y_t$ é o rótulo correspondente. Agora pode ser feita uma definição formal de transferência de aprendizado:

Definição 3 *(Transferência de aprendizado). Dado um domínio de origem* D_s *e uma tarefa de aprendizagem* T_s*, um domínio de destino* D_t *e uma tarefa de aprendizagem* T_t*, a transferência de aprendizado corresponde a métodos que têm como objetivo melhorar a aprendizagem da função de predição* $f_t(\cdot)$ *no domínio de destino* D_t *usando o conhecimento disponível* D_s *e* T_s*, onde* $D_s \neq D_t$ *ou* $T_s \neq T_t$*.*

2.5.2 Transferência de aprendizado com redes convolutivas profundas

A transferência de aprendizado utilizando redes convolutivas tem sido aplicada com sucesso em vários domínios e tarefas de visão computacional tais como análise de imagens médicas [Christodoulidis et al., 2017, Elmahdy et al., 2017, Tajbakhsh et al., 2016], classificação de imagens de comida [Heravi et al., 2017], análise de imagens de raio-x [Akçay et al., 2016], detecção de placas de trânsito em vídeos [Changzhen et al., 2016], entre muitas outras, obtendo, na grande maioria dos casos, resultados que representam o estado da arte para o problema.

Na literatura, há duas maneiras principais de se utilizar as redes neurais convolutivas profundas para transferência de representação: como extrator de características para outro algoritmo de aprendizagem supervisionado ou como inicialização dos pesos da arquitetura para o treinamento em uma nova tarefa. Uma terceira abordagem consiste em treinar redes convolutivas que aprendem tarefas distintas ao mesmo tempo, o que também é chamado de aprendizado multitarefa.

2.5.3 Redes neurais convolutivas como extratores de características

Diversos estudos mostram que características genéricas extraídas de redes convolutivas profundas funcionam como excelentes descritores de imagem para abordar uma variedade imensa de tarefas de reconhecimento, tais como classificação de objetos, análise de imagens médicas, análise de imagens de satélite, dentre muitas outras [Donahue et al., 2014, Oquab et al., 2014].

Nessa abordagem, utiliza-se as saídas de uma das camadas da rede convolutiva previamente treinada como características de entrada para algum algoritmo de aprendizagem. Dessa forma, a rede não passa por um novo processo de treinamento no novo domínio ou tarefa, agindo como uma caixa-preta entre a imagem original e as características obtidas na sua saída. O treinamento é feito exclusivamente no algoritmo de aprendizagem cujas entradas são as características de alto nível produzidas pela rede convolutiva. A Figura 2.7 apresenta um esquema da utilização de uma rede convolutiva pré-treinada na base de dados ImageNet como extrator de características, onde o classificador é uma rede neural completamente conectada com duas camadas.

2.5.4 Transferência de aprendizado como pré-treinamento supervisionado

Apesar de diversos estudos mostrarem que as redes convolutivas profundas treinadas em bases genéricas são excelentes extratores de características, permitir o retreinamento de qualquer camada para o novo problema é outra opção frequentemente utilizada na literatura. Nesses casos, além de retreinar a camada de classificação no novo domínio de imagens, o treinamento é propagado para qualquer camada desejada, o que é chamado de *fine tuning*. Essa abordagem permite que a rede aprenda a representar características de alto nível específicas para o novo domínio ou nova tarefa [Yosinski et al., 2014].

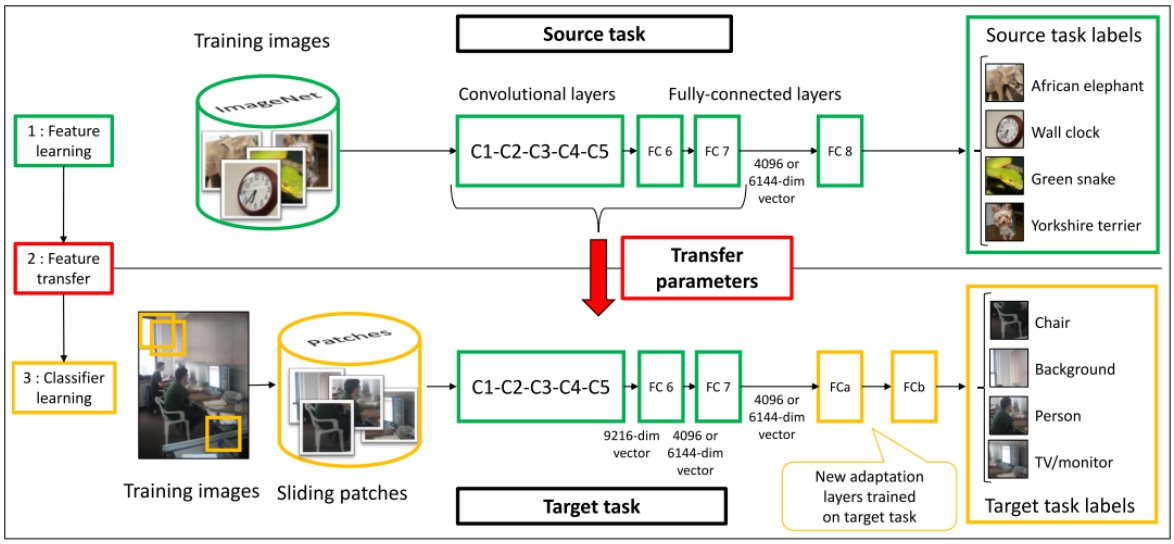


Figura 2.7: Esquema da utilização de uma rede neural convolutiva como extrator de características, realizando transferência de representação.

Fonte: Oquab et al. [2014].

Nesse tipo de técnica, a transferência de representação atua como inicialização dos pesos das camadas da rede para o treinamento do modelo na nova tarefa. Em modelos que são bastante sensíveis à inicialização dos pesos como redes convolutivas profundas, esse tipo de estratégia é essencial para a obtenção de bons resultados.

2.5.5 Aprendizado multitarefa

Uma abordagem mais recente de transferência de aprendizado com redes neurais convolutivas consiste em misturar as abordagens anteriores com treinamento de múltiplas tarefas na mesma estrutura. Neste modelo, a tarefa de destino e outras tarefas relacionadas são treinadas conjuntamente com camadas de entrada e intermediárias compartilhadas e camadas de saída separadas. Nesse caso, a transferência de aprendizado se dá entre as tarefas distintas.

2.6 Algoritmos genéticos

Algoritmos genéticos são técnicas de otimização com inspiração na biologia evolutiva para achar soluções aproximadas em problemas de otimização e busca. Entretanto, segundo Mitchell [1998], não existe uma definição rigorosa aceita por todos na comunidade de computação evolucionária que diferencie algoritmos genéticos de outros métodos evolucionários. Contudo, a maioria dos métodos chamados de algoritmo genético

tem pelo menos os seguintes elementos em comum: uma população de cromossomos (também chamadados de indivíduos), seleção de indivíduos de acordo com uma função-objetivo, cruzamento para produção de uma nova geração e mutação aleatória da nova geração.

Cada indivíduo numa população pode ser interpretado como um ponto no espaço de busca das soluções candidatas para um problema. O algoritmo genético processa as populações, substituindo uma população por outra sucessivamente. Geralmente, uma função-objetivo é utilizada para calcular a aptidão dos indivíduos, indicando quão bem cada um deles resolve o problema em questão [Mitchell, 1998].

Dada uma definição clara do problema a ser resolvido e uma representação de soluções candidatas, um algoritmo genético simples funciona da seguinte forma, segundo Mitchell [1998]:

1. Crie uma população aleatória de n indivíduos (soluções candidatas ao problema);
2. Calcule a aptidão de cada indivíduo x na população utilizando a função-objetivo;
3. Repita os seguintes passos para criar n novos indivíduos:
    a) Selecione um par de indivíduos da população atual, onde a probabilidade de seleção de um indivíduo é uma função de sua aptidão;
    b) Com uma probabilidade p_c (prababilidade de cruzamento), faça o cruzamento do par em um ponto ou mais pontos para gerar um novo indivíduo. Caso o indivíduo gerado não seja uma solução candidata, retorne uma cópia de um dos pais;
    c) Faça uma mutação aleatória com probabilidade p_m (probabilidade de mutação) e adicione o indivíduo resultante na nova população.
4. Substitua a população atual pela nova população;
5. Retorne ao passo 2.

Cada iteração entre os passos 2 e 5 é chamada de geração. Normalmente, um algoritmo genético é iterado de 50 a 500 ou mais gerações e, ao final dessa execução, geralmente há um ou mais indivíduos que são boas soluções aproximadas do problema.

2.7 Métricas de avaliação

Em aprendizagem de máquina há diversas formas de se avaliar o desempenho dos classificadores. As métricas referentes à qualidade da classificação são obtidas a partir

da matriz de confusão que registra a quantidade de intâncias correta e incorretamente classificadas para cada classe [Sokolova et al., 2006]. A Tabela 2.1 apresenta a matriz de confusão para um problema binário, indicando os verdadeiros positivos (tp), falsos negativos (fn), falsos positivos (fp) e verdadeiros negativos (tn).

Tabela 2.1: Matriz de confusão para classificação binária.

		Classe prevista	
		Positiva	Negativa
Classe real	Positiva	tp	fn
	Negativa	fp	tn

A acurácia é uma métrica que avalia o desempenho dos modelos sem focar em uma classe específica, não fazendo, portanto, distinção entre o número de rótulos corretos de classes diferentes, sendo definida por:

$$Acurácia = \frac{tp + tn}{tp + fp + fn + tn} \tag{2.1}$$

Para problemas multiclasse, a acurácia é calculada dividindo-se a soma da diagonal principal da matriz de confusão pela quantidade total de instâncias de treinamento.

Em aprendizado profundo é comum utilizar-se a abordagem top-N, onde uma predição é considerada correta se a resposta esperada estiver entre as N respostas com maior probabilidade fornecidas pelo modelo. Assim, constuma-se avaliar os modelos em termos de acurácia top-1 (acurácia convencional) e acurácia top-5.

Para avaliar o desempenho de cada classe, a partir da matriz pode-se obter a precisão e a revocação. A revocação indica a razão de instâncias de uma dada classe que foram corretamente classificadas. A precisão indica a razão de instâncias classificadas corretamente dentre as que foram classificadas em uma dada classe. O F1-score é a média harmônica entre precisão e revocação e indica uma relação de compromisso entre essas duas métricas. Essas três métricas são descritas pelas seguintes equações:

$$Precisão = \frac{tp}{tp + fp} \tag{2.2}$$

$$Revocação = \frac{tp}{tp + fn} \tag{2.3}$$

$$F1\text{-}score = \frac{2 * Precisão * Revocação}{Precisão + Revocação} \tag{2.4}$$

2.8 Considerações Finais

Neste capítulo foram apresentados os fundamentos das diversas arquiteturas de redes neurais profundas, estratégias de treinamento das mesmas e métricas de avaliação. Também foram adicionadas explicações sobre as diferentes formas de se fazer transferência de aprendizado a partir de modelos de redes profundas treinados em outros domínios para o domínio alvo. Além disso, foi incluída uma descrição sobre algoritmos genéticos que foram utilizados nas abordagens de particionamento propostas neste trabalho.

Esses conceitos são essenciais para a compreensão do capítulo de trabalhos relacionados (Capítulo 3), no qual será discutido como redes profundas têm sido aplicadas ao problema de classificação em imagens de armadilhas fotográficas.

3

Trabalhos Relacionados

Neste capítulo é apresentado um resumo das soluções existentes para o problema de classificação de imagens de armadilhas fotográficas. Os trabalhos revisados abordam o problema de classificação de espécies animais sob diversas condições e restrições quanto aos tipos de imagens a serem utilizadas. A Seção 3.1 apresenta os trabalhos que buscam classificar as espécies automaticamente. Na Seção 3.2 é analisada a forma de particionamento das bases de imagens utilizadas nos trabalhos correlatos. A Seção 3.3 descreve as abordagens para transferência de aprendizado para bases de armadilhas fotográficas utilizadas nos trabalhos que aplicam esta técnica. Por fim, uma síntese comparativa desses trabalhos é feita na Seção 3.4.

3.1 Classificação de imagens de armadilhas fotográficas

Diversos trabalhos na literatura têm abordado o problema de classificação automática de animais em imagens de armadilhas com o objetivo de tornar mais eficiente a extração de informações dessas bases, reduzindo ou eliminando a necessidade de análise manual das imagens. Todos utilizam alguma técnica de aprendizagem de máquina para treinar um modelo capaz de realizar a classificação.

No trabalho apresentado por Yu et al. [2013], foi utilizada uma versão adaptada do método *Sparse Coding Spatial Pyramid Matching* (ScSPM) para fazer o reconhecimento de espécies em imagens de armadilhas fotográficas. Nessa versão do método, foi aplicada para a extração de características locais uma combinação entre as técnicas *Scale Invariant Feature Transform* (SIFT) e *Cell Structured Local Binary Patterns* (cLBP) para representar o objeto de interesse. Em seguida, a partir dessas características locais foram geradas características globais utilizando-se as técnicas *Weighted*

Sparse Coding e *Multiscale Pyramid Kernel*. Por fim, foi utilizado o classificador *Support Vector Machine* (SVM) com *kernel* linear para classificar as espécies nas imagens. Os experimentos foram realizados em uma base com mais de 7000 imagens de armadilhas fotográficas de 18 espécies animais diferentes, atingindo uma acurácia média de 82% na classificação da espécie. Vale ressaltar que, nesse trabalho, inicialmente as imagens foram recortadas manualmente em retângulos justos em torno dos animais.

Chen et al. [2014] propõem um método totalmente automático para reconhecimento de espécies em imagens de armadilhas fotográficas. Inicialmente as imagens são segmentadas para remover o plano de fundo utilizando-se o algoritmo *Ensemble Video Object Cut* (EVOC). Em seguida, a classificação de espécies é realizada por uma rede neural convolutiva composta por três camadas convolutivas e três camadas de *pooling*. Para avaliação do método proposto foi utilizada uma base contendo 14346 imagens para treinamento e 9530 imagens para teste de 20 diferentes espécies comuns na América do Norte. Nesse trabalho, atingiu-se uma acurácia de apenas 38% na classificação, mas que foi superior aos 33% obtidos utilizando-se a técnica *bag-of-words* usada como referência.

No trabalho de Villa et al. [2017] são feitos vários experimentos para comparar o desempenho de arquiteturas consagradas de redes neurais profundas para o problema de classificação de imagens de armadilhas fotográficas. Além disso, também foram avaliadas diversas configurações de base de dados utilizadas no treinamento: base com classes desbalanceadas, base com classes balanceadas, base contendo imagens em que os animais sempre aparecem em primeiro plano e uma base em que os animais foram segmentados manualmente. Para a preparação das bases, foram selecionadas imagens de alta qualidade de 26 espécies animais obtidas com armadilhas fotográficas instaladas no Parque Nacional do Serengeti. Foram testadas as arquiteturas AlexNet, VGGNet, GoogLenet e ResNet (versões com 50, 101 e 152 camadas), sendo que a arquitetura ResNet101 obteve melhor resultado, atingindo uma taxa de acurácia de 88,9% na base em que os animais foram segmentados manualmente. Para o treinamento foi utilizada transferência de aprendizado, sendo todas as arquiteturas previamente treinadas na base de dados ImageNet. Os experimentos também mostraram que uma base com classes desbalanceadas apresentou resultados significativamente inferiores aos resultados obtidos com a base balanceada.

Em outro trabalho, Villa et al. [2016] testaram as arquiteturas AlexNet, VGGNet, GoogLenet e ResNet em uma base de dados com imagens de baixa qualidade obtidas no território colombiano. Todas as imagens foram previamente segmentadas manualmente por especialistas, os quais recortaram os animais do plano de fundo. Nesse trabalho, no entanto, buscou-se classificar os animais em classes taxonômicas de nível mais alto

e não em espécies. Assim, foram realizados experimentos para classificar animais como aves ou mamíferos (1572 imagens) e experimentos para classificar os mamíferos em duas classes (2597 imagens). Também nesse trabalho a arquitetura ResNet101 obteve os melhores resultados, atingindo 97,5% de taxa de acerto na classificação entre aves e mamíferos e 90,23% na classificação dos mamíferos.

Norouzzadeh et al. [2018] treinaram as arquiteturas de redes profundas utilizando uma abordagem de aprendizado multitarefa, onde o modelo foi treinado simultaneamente para as tarefas de reconhecimento de espécies, contagem de indivíduos e identificação da atividade do animal, tratando essas tarefas como sendo de classificação. Nesse trabalho, os autores partiram do princípio de que a base Snapshot Serengeti possui uma quantidade suficiente de imagens para treinar redes profundas do zero e, especificamente, para a tarefa de classificação de espécies, os autores obtiveram resultados significativamente superiores em relação aos de Villa et al. [2017], atingindo 93,8% de acurácia de top-1 com a ResNet101 e ResNet152. Nesse trabalho também são treinados modelos para detectar se uma imagem contém ou não um animal, atingindo 96,6% de taxa de acertos, o que é similar à taxa de acertos da comunidade de voluntários responsável por rotular manualmente a base Snapshot Serengeti. Por fim, o trabalho também faz um classificação do evento de captura, no entanto, a classificação se baseia simplesmente numa média estatística da classificação de cada imagem da sequência, sem utilizar um método de aprendizagem para tal tarefa.

Willi et al. [2018] utilizaram a arquitetura ResNet18 para classificar imagens em diferentes espécies de animais e para classificar em animal, veículos ou plano de fundo. A arquitetura ResNet18 foi escolhida pelos autores para balancear acurácia e custo computacional tendo como referência os resultados de Norouzzadeh et al. [2018], onde a ResNet18 obteve acurácia top-1 apenas 0,5% menor que a ResNet101, o modelo com melhor desempenho. Foram treinados modelos diferentes para cada uma das quatro bases de imagens de armadilhas fotográficas utilizadas: uma versão ampliada da Snapshot Serengeti com 7,3 milhões de imagens, sendo 1,2 milhão de imagens de animais; Camera CATalogue com 520 mil imagens, 140 mil com animais; Elephant Expedition com 420 mil imagens, 50 mil com animais; e Snapshot Wisconsin com 500 mil imagens, 300 mil de animais. A acurácia para identificação de animais variou entre 88,7% e 92,7%, enquanto para a classificação entre animal e plano de fundo, a acurácia ficou entre 91,2% e 98%. Os autores também fazem um experimento em que combinam as predições dos modelos com as classificações manuais realizadas por voluntários, atingindo uma redução de 43% na quantidade de esforço humano necessário para rotular novas imagens.

No trabalho de Tabak et al. [2018], a arquitetura ResNet18 foi treinada do zero

para classificar animais em 27 classes, entre elas a classe plano de fundo, em uma base com 3,7 milhões de imagens obtidas em treze projetos de armadilhas fotográficas diferentes ao longo de cinco estados norte-americanos. Foi obtida uma acurácia top-1 de 97,6% e uma acurácia top-5 de 99,9%. Os autores fizeram uma avaliação *out-of-sample* em um conjunto de teste de imagens capturadas no Canadá, atingindo uma acurácia top-1 de 82%. Entretanto, esse conjunto de teste contém apenas imagens de 4 espécies, sendo 3 delas majoritárias, representando cerca de 68% das instâncias da base utilizada para treinamento.

3.2 Abordagens utilizadas para o particionamento de bases

Grande parte dos trabalhos apresentados na Seção 3.1 não faz nenhum tratamento durante o particionamento das bases para evitar que imagens muito similares às do treinamento sejam utilizadas durante o teste, fazendo a divisão em treino e teste de maneira completamente aleatória [Chen et al., 2014, Tabak et al., 2018, Villa et al., 2016, 2017, Yu et al., 2013].

Para minimizar o problema de imagens similares no treino e no teste, Norouzzadeh et al. [2018] realizaram um particionamento colocando imagens do mesmo evento de captura na mesma partição. Essa estratégia também foi adotada por Willi et al. [2018] que, além disso, mantiveram juntos na mesma partição as imagens de todos os eventos de captura ocorridos dentro de uma janela de 30 minutos. Essa janela temporal foi adotada por ser comum que um mesmo indivíduo dispare vários eventos de captura em sequência. Nessa situação, as imagens costumam ser muito similares, mesmo que seja programado um intervalo de tempo mínimo entre os eventos de captura.

Em outros domínios, diversas abordagens são adotadas para evitar sobreposição de imagens similares no treino e no teste. No particionamento da base de imagens de objetos Microsoft COCO [Lin et al., 2014], instâncias obtidas na mesma data ou capturadas pelo mesmo fotográfico são mantidas na mesma partição. Na base iNat2017 [Horn et al., 2018], que contém imagens de seres vivos, as instâncias enviadas ao projeto iNaturalist por cada colaborador são mantidas na mesma partição, a fim de que o comportamento particular de um usuário, como plano de fundo, equipamento fotográfico ou localização, não seja fonte de informação útil para classificar um determinado grupo taxonômico.

3.3 Transferência de aprendizado para bases de armadilhas fotográficas

Nos trabalhos de Villa et al. [2016, 2017], as arquiteturas utilizadas são pré-treinadas na ImageNet, sendo utilizadas tanto como extratores de características com o treinamento somente da última camada, quanto com a realização de *fine tuning* das camadas convolutivas. Essa estratégia foi utilizada porque os modelos pré-treinados na ImageNet normalmente aprendem características com alta capacidade de representação que funcionam como excelentes descritores de imagem para as mais diversas tarefas de reconhecimento visual.

Por outro lado, Norouzzadeh et al. [2018] sugerem que modelos pré-treinados na base Snapshot Serengeti podem produzir resultados melhores do que os modelos pré-treinados na ImageNet ao se fazer transferência de aprendizado para outras bases de armadilhas fotográficas de menor escala, pois as classes entre projetos desse tipo seriam mais similares entre si do que com as classes da ImageNet. No entanto, não foi realizado um estudo comparativo para testar essa hipótese devido os autores não terem acesso a outras bases de armadilhas fotográficas.

No trabalho de Willi et al. [2018], foi realizado um estudo para comparar o desempenho entre treinar modelos do zero para cada uma das bases e fazer transferência de aprendizado, tendo como ponto de partida modelos treinados do zero na Snapshot Serengeti. Foi verificado um acréscimo de até 10,3% na acurácia dos modelos quando foi realizada a transferência de aprendizado. No entanto, não foi realizada uma comparação com transferência a partir da ImageNet, assumindo-se que transferir aprendizado de outras bases de armadilhas fotográficas produz resultados melhores.

3.4 Síntese dos trabalhos relacionados

A segmentação manual realizada nos trabalhos de Villa et al. [2016, 2017], Yu et al. [2013] é um cenário que não escala como solução prática e, além disso, modelos de aprendizado profundo levam em consideração também o contexto. Tendo isso em vista, para os experimentos desta dissertação serão utilizadas imagens completas, sem segmentação explícita.

Enquanto boa parte dos trabalhos ignora a similaridade das imagens de armadilhas fotográficas e faz um particionamento completamente aleatório, os trabalhos de Norouzzadeh et al. [2018] e de Willi et al. [2018] utilizam um particionamento baseado nos eventos de captura para lidar o viés otimista do conjunto de teste. Partindo

da hipótese de que mesmo imagens de eventos diferentes obtidos em datas próximas ainda são muito similares, o presente trabalho investiga outras abordagens de particionamento, como por pontos de captura, por tempo e por pontos de captura por classe, comparando-as com o particionamento por evento, a fim de obter um conjunto de teste que ofereça condições de avaliação mais próximas às situações reais de utilização dos modelos.

Quanto à transferência de aprendizado para imagens de armadilhas fotográficas, alguns trabalhos aplicam essa técnica a partir da base ImageNet [Villa et al., 2016, 2017], enquanto outros argumentam que a transferência a partir de outras bases de armadilhas fotográficas é melhor [Norouzzadeh et al., 2018, Willi et al., 2018]. No entanto, não foi realizado nenhum estudo comparativo para comprovar qual modelo-base oferece melhores condições para realizar a transferência, o que será abordado no presente trabalho. Além disso, o viés otimista dos conjuntos de teste pode mascarar o real desempenho dos modelos que utilizam transferência de aprendizado, apresentando resultados bons devido aos problemas de particionamento dos dados mencionados anteriormente.

A Tabela 3.1 apresenta um resumo comparativo dos trabalhos discutidos neste capítulo.

3.5 Considerações finais

Este capítulo apresentou uma síntese dos estudos que propõem soluções para classificação automática de animais em imagens de armadilhas fotográficas, e realizou uma comparação em relação ao presente trabalho. A partir da análise comparativa pôde ser observado que os trabalhos que utilizam aprendizado profundo obtêm resultados muito superiores em relação aos que utilizam técnicas tradicionais. Foi possível perceber também uma preocupação dos autores em aplicar redes neurais profundas em projetos com bases de imagens de menor escala. No entanto, há uma certa indiferença quanto à produção de conjuntos de teste *out-of-sample* que efetivamente forneçam condições de avaliação mais próximas de situações reais de utilização dos modelos.

Tabela 3.1: Síntese comparativa dos trabalhos relacionados

Trabalho	Processamento manual	Modelo	Transferência de aprendizado	Particionamento da base	Resultados
Yu et al. [2013]	Sim	SVM	Não	Aleatório	82%
Chen et al. [2014]	Não	Rede convolutiva personalizada	Não	Aleatório	38%
Villa et al. [2017]	Sim[1]	AlexNet, VGGNet, GoogLenet e ResNet	Sim	Aleatório	57% – 88,9%
Villa et al. [2016]	Sim	AlexNet, VGGNet, GoogLenet e ResNet	Sim	Aleatório	90,23% – 97,5%
Norouzzadeh et al. [2018]	Não	AlexNet, VGGNet, GoogLenet e ResNet	Não	Evento	93,8%
Willi et al. [2018]	Não	ResNet18	Sim	Evento[2]	88,7% – 92,7%
Tabak et al. [2018]	Não	ResNet18	Não	Aleatório	82% – 97,6%

[1] Inclui tanto experimentos sem nenhum pré-processamento das imagens quanto experimentos em que os animais foram segmentados manualmente.
[2] Adicionalmente, mantém-se na mesma partição eventos de captura que foram obtidos dentro de uma janela de 30 minutos.

4

Estratégias de avaliação *out-of-sample* de modelos para classificação de animais em imagens de armadilhas fotográficas

Este capítulo apresenta estratégias para avaliação *out-of-sample* de modelos para o problema de classificação de animais em imagens de armadilhas fotográficas. Inicialmente é feita uma análise baseada na forma de construção das bases de imagens de armadilhas fotográficas a fim de identificar situações que podem representar condições mais justas de avaliação dos modelos, reduzindo o viés otimista do conjunto de teste (Seção 4.1). Em seguida, as condições identificadas são avaliadas experimentalmente: 1) instâncias de pontos de captura não incluídos no treinamento (Seção 4.2), 2) instâncias obtidas em data posterior às utilizadas no treinamento (Seção 4.3) e 3) instâncias de classes que não foram vistas durante o treinamento em determinados pontos de captura (Seção 4.4). Finalmente, são apresentadas recomendações para a avaliação *out-of-sample* de modelos de acordo com o protocolo utilizado pelo projeto de armadilhas fotográficas.

4.1 Condições de predição *out-of-sample* de animais em imagens de armadilhas fotográficas

Em projetos de armadilhas fotográficas, as câmeras são distribuídas em uma determinada região de acordo com um protocolo que pode variar conforme os objetivos do monitoramento da vida selvagem que se pretende fazer. Esse protocolo indica os proce-

dimentos a serem seguidos durante todas as fases do projeto, tais como planejamento, instalação das câmeras e extração de informações das imagens.

Portanto, o protocolo adotado estabelece as diretrizes para escolha dos locais (pontos de captura) onde as câmeras serão fixadas, bem como o tempo de coleta, que pode ser contínuo. Além disso, geralmente o protocolo determina que as câmeras devam ser programadas para obter mais de uma imagem a cada vez que o sensor de movimento for acionado (evento de captura). Dessa forma, há uma probabilidade maior de capturar uma imagem que permita identificar o animal na cena. Consequentemente, as bases resultantes contêm imagens muito similares entre si, seja por pertencerem ao mesmo evento de captura, seja por possuírem o mesmo plano de fundo, considerando que foram obtidas no mesmo ponto de captura.

Essa semelhança entre as imagens pode permitir que os modelos treinados nessas bases identifiquem melhor os animais nessas imagens. No entanto, modelos de aprendizado profundo têm alta capacidade de representação, podendo facilmente memorizar toda a base de treinamento mesmo que as instâncias possuam rótulos completamente aleatórios, conforme demonstrado por Zhang et al. [2016]. Assim, há a possibilidade dos modelos fazerem associações entre os rótulos e outros elementos das imagens que não sejam necessariamente os animais a serem identificados. Nesse caso, se a base de teste for construída sem levar em consideração a similaridade das imagens de armadilhas fotográficas, a avaliação pode superestimar a capacidade de generalização dos modelos.

Conforme mencionado em capítulos anteriores, muitos trabalhos que investigam classificação de imagens de armadilhas fotográficas têm ignorado essa similaridade entre as imagens da base, pois realizam o particionamento entre treino e teste de maneira completamente aleatória, como em Chen et al. [2014], Tabak et al. [2018], Villa et al. [2017] e Yu et al. [2013]. Outros utilizam a abordagem de manter na mesma partição as imagens de um mesmo evento de captura [Norouzzadeh et al., 2018, Willi et al., 2018]. No entanto, esse tipo de avaliação não permite verificar se o modelo é robusto o suficiente para identificar animais em situações que fogem à distribuição presente na base de treinamento, tais como a mudança natural da vegetação no plano de fundo ao longo tempo ou a adição de novos pontos de captura ao projeto. Sendo assim, para identificar e analisar essas situações *out-of-sample* que podem ocorrer durante a utilização dos modelos em condições reais, foi realizado neste trabalho um estudo de caso na base de imagens do projeto de armadilhas fotográficas Snapshot Serengeti [Swanson et al., 2015a], conforme descrito a seguir.

4.1.1 Base de imagens Snapshot Serengeti

O projeto Snapshot Serengeti é um dos maiores projetos de armadilhas fotográficas do mundo, contendo 225 pontos de captura espalhados pelo Parque Nacional do Serengeti na Tanzânia [Swanson et al., 2015b]. Esse projeto conta com a colaboração de voluntários que fazem a extração manual de dados das imagens obtidas, identificando, por exemplo, as espécies presentes, contagem de indivíduos e atividade do animal (correndo, comendo, descansando, etc.). Os dados são disponibilizados publicamente para que a comunidade possa estudar a ecologia da região.

Para o contexto deste trabalho, foi utilizado somente um subconjunto contendo as 26 classes com mais de mil imagens coletadas entre junho de 2010 e maio de 2013, conforme especificado no trabalho de Villa et al. [2017], sendo denominado de S26. Esse subconjunto inclui somente instâncias com uma única espécie presente na imagem, sendo, portanto, eliminadas as imagens sem animal ou com mais de uma espécie. As quantidades totais de instâncias por classe (ver Tabela 4.1), no entanto, não são iguais às utilizadas por Villa et al. [2017] devido à indisponibilidade de algumas imagens.

Tabela 4.1: Quantidade de imagens por classe da base de dados S26.

Fonte: Elaborado pelo autor.

Classe	# Instâncias	Classe	# Instâncias
wildebeest	243404	lionFemale	8389
zebra	147668	eland	6683
gazelleThomsons	112214	topi	5806
hartebeest	33853	baboon	4416
buffalo	33141	reedbuck	4109
human	26245	cheetah	3330
elephant	25094	dikDik	3325
giraffe	21735	hippopotamus	3228
impala	21592	lionMale	2149
guineaFowl	21587	ostrich	1870
warthog	20781	koriBustard	1865
gazelleGrants	20250	secretaryBird	1209
hyenaSpotted	10119	jackal	1154
		Total	**785216**

A base S26 possui grande desbalanceamento entre as classes, apresentando uma distribuição de cauda longa, isto é, algumas poucas classes concentram uma parcela significativa do total de instâncias da base, enquanto uma grande quantidade de clas-

ses apresenta um número de instâncias proporcionalmente menor, conforme pode ser visualizado na Figura 4.1. Essa distribuição é esperada para bases de imagens de armadilhas fotográficas, uma vez que as observações de seres vivos costumam seguir essa distribuição [Joly et al., 2014]. Especificamente na S26, as três classes majoritárias apresentam mais de 100 mil imagens cada e juntas representam aproximadamente 65% do total da base. Isso deve ser levado em consideração durante a avaliação do desempenho de modelos neste tipo de problema, pois se um modelo aprender somente essas três classes majoritárias, iniciará com 65% de acurácia, mas isso não significa que será efetivamente bom se errar as demais classes de interesse do problema.

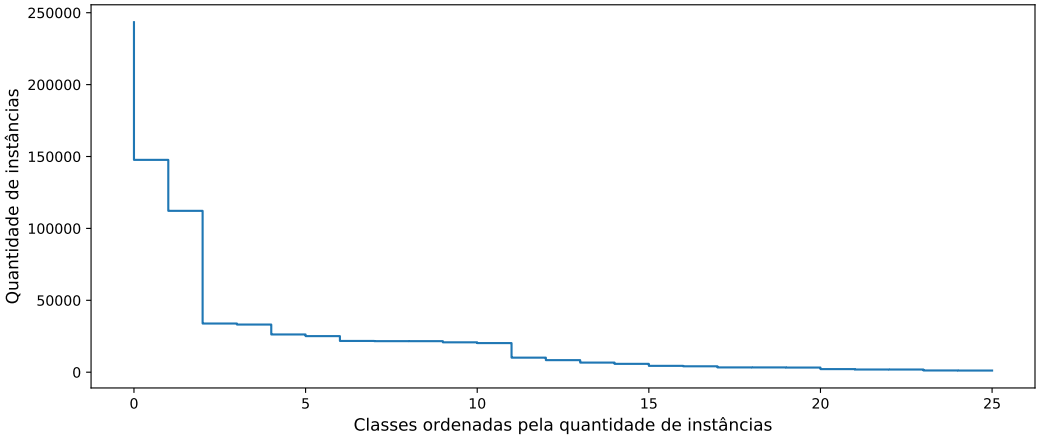


Figura 4.1: Distribuição das instâncias por classe na base S26.

Fonte: Elaborado pelo autor.

Além de haver um desbalanceamento interclasse, há ainda um desbalanceamento na quantidade de imagens obtidas por ponto de captura (Figura 4.2), com mais imagens obtidas em determinados pontos da rede. A distribuição das instâncias das classes também varia dentre os pontos de captura, com algumas espécies sendo mais frequentes em determinados locais e menos em outros, como pode ser observado na Figura 4.3 que apresenta a distribuição das classes para alguns pontos de captura da S26.

4.1.2 Análise de condições *out-of-sample*

Um fator observado no projeto Snapshot Serengeti é a expansão do número de pontos de captura, com a adição de 25 novos locais monitorados a partir de fevereiro de 2012 [Swanson et al., 2015b]. De fato, a automatização da extração de informações das imagens coletadas pode permitir e incentivar mais ainda a expansão de projetos. Assim, faz-se necessária uma avaliação mais cuidadosa do desempenho de modelos treinados anteriormente ao serem aplicados em imagens de pontos de captura não incluídos no

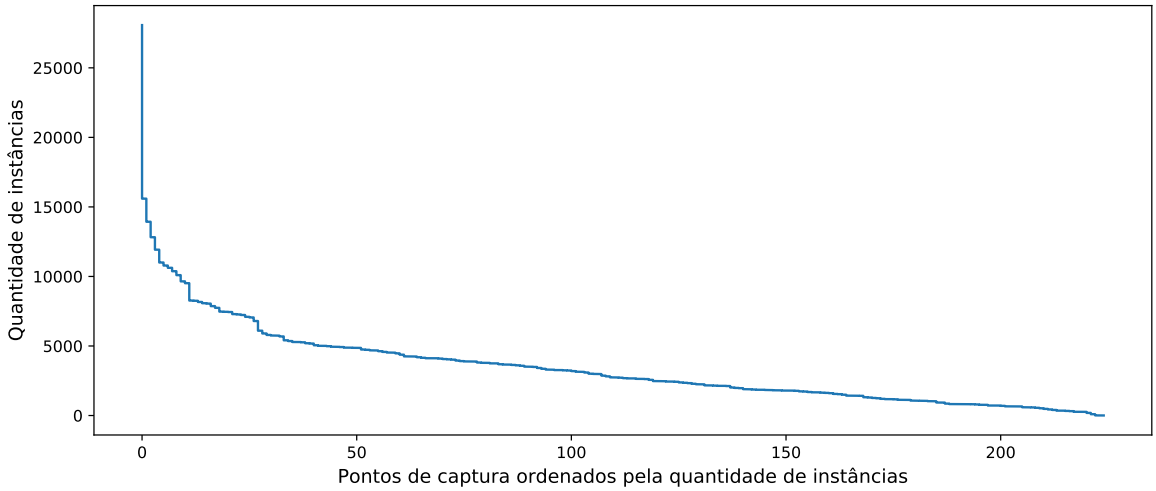
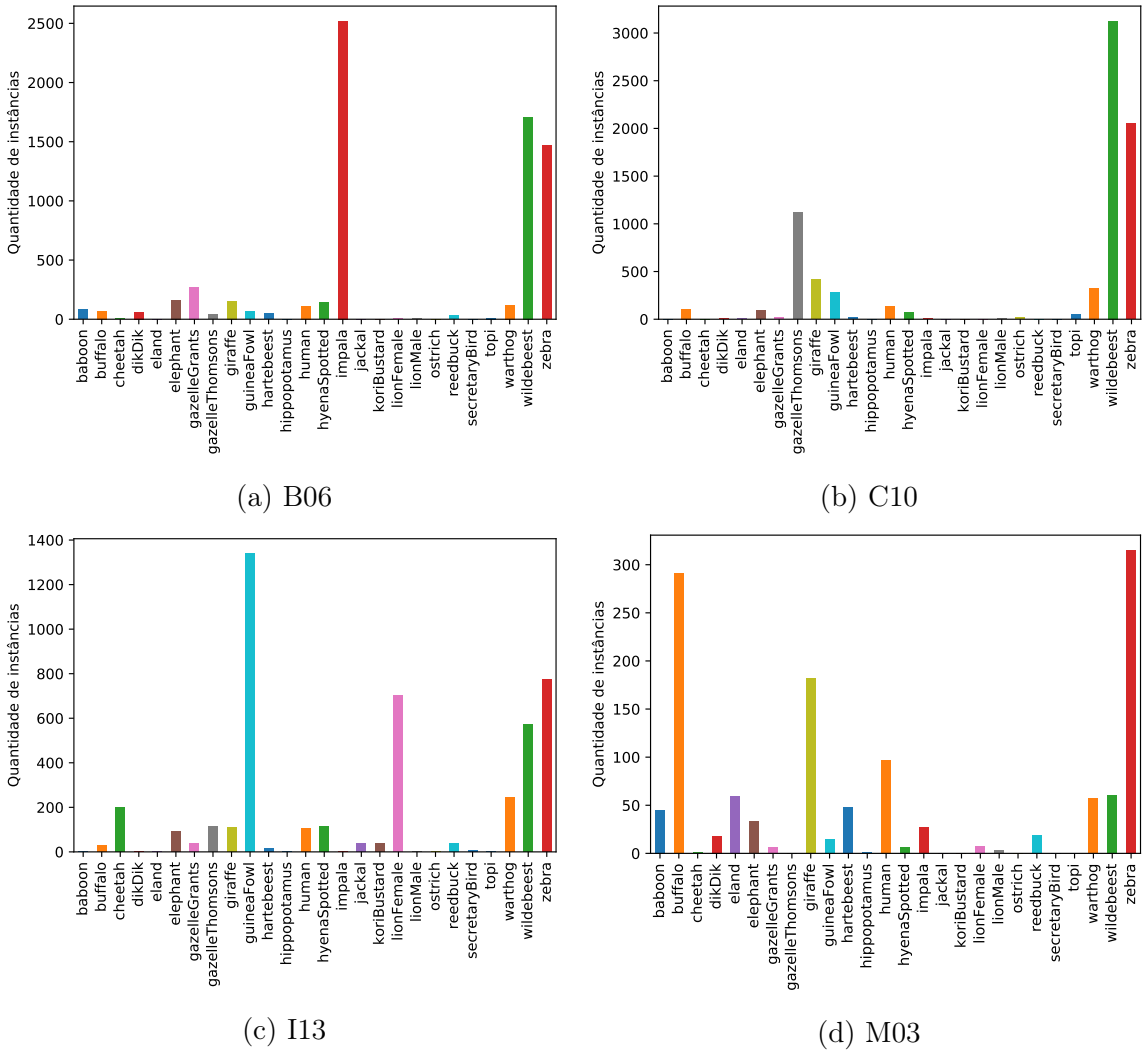


Figura 4.2: Distribuição das instâncias por ponto de captura na base S26.

Fonte: Elaborado pelo autor.

treinamento para garantir a confiabilidade das predições. Isso ocorre porque o modelo pode fazer *overfitting* no plano de fundo dos pontos de captura utilizados no treinamento. Uma forma usual de minimizar esse problema é a aplicação de aumento de dados na base, mas, como não é possível saber o local exato do animal na imagem, essa técnica deve ser aplicada de forma moderada, para evitar cortar o animal da imagem e fornecer ao modelo uma instância que contenha somente o plano de fundo, fato que pode prejudicar ainda mais a aprendizagem. Nesse caso, uma forma de avaliar o modelo, dado o viés com relação ao plano de fundo dos pontos de captura, é fazer o particionamento para treinamento e teste mantendo-se as imagens do mesmo ponto de captura na mesma partição.

Em contrapartida, se o modelo treinado for utilizado para identificar animais em imagens coletadas somente na rede de câmeras utilizada no treinamento, espera-se que o desempenho se mantenha razoável, mesmo que haja um viés com relação ao plano de fundo. Entretanto, ainda que não haja adição de novos pontos de captura e que o protocolo especifique que a câmera deva ser instalada sempre na mesma posição a cada manutenção, haverá fatores que não dependem do protocolo, mas que podem alterar as condições de plano de fundo, como a mudança sazonal da vegetação ao longo do tempo. A Figura 4.4 exibe um conjunto de imagens de um mesmo ponto de captura ao longo do período de coleta da base S26, onde se pode observar essa mudança da vegetação, bem como uma certa modificação do ângulo de visão da câmera. Contudo, espera-se que os modelos sejam robustos o suficiente para lidar com isso. Porém, devido à similaridade das imagens de armadilhas fotográficas, uma forma mais adequada de avaliar o modelo



(a) B06

(b) C10

(c) I13

(d) M03

Figura 4.3: Distribuição das espécies em quatro pontos de captura da base S26: (a) B06, (b) C10, (c) I13 e (d) M03.

Fonte: Elaborado pelo autor.

considerando essas mudanças se dá por meio do particionamento por tempo, isto é, treina-se o modelo com imagens até uma determinada data e testa-se com imagens obtidas posteriormente. Essa forma de avaliação se aproxima das condições reais de utilização do modelo ao ser aplicado em novas imagens.

Uma terceira condição refere-se à capacidade do modelo em identificar classes em locais em que não foram observadas durante o treinamento. A distribuição de cada classe não é uniforme em relação aos pontos de captura, havendo locais com maior concentração de observações, como mostra a Figura 4.5. Nesse caso, se o modelo aprender a associar uma classe a um local no qual esta seja majoritária, um particionamento somente por evento de captura não seria capaz de evidenciar esse viés, uma vez que

Figura 4.4: Variação das imagens obtidas no ponto de captura F02 ao longo do período de coleta.

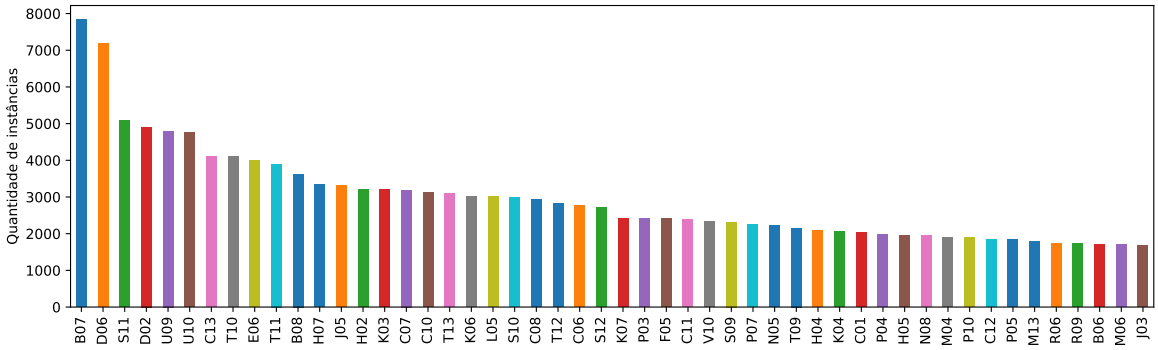
Fonte: Base de Imagens Snapshot Serengeti [Swanson et al., 2015a].

haveria muitos eventos do animal no mesmo local, tanto no treino quanto no teste. A Figura 4.6 mostra que, para a maioria das classes, há uma quantidade significativa de locais em que elas não foram observadas. Dada essa condição, espera-se que o modelo seja capaz de identificar a classe independentemente do local em que ela apareça.

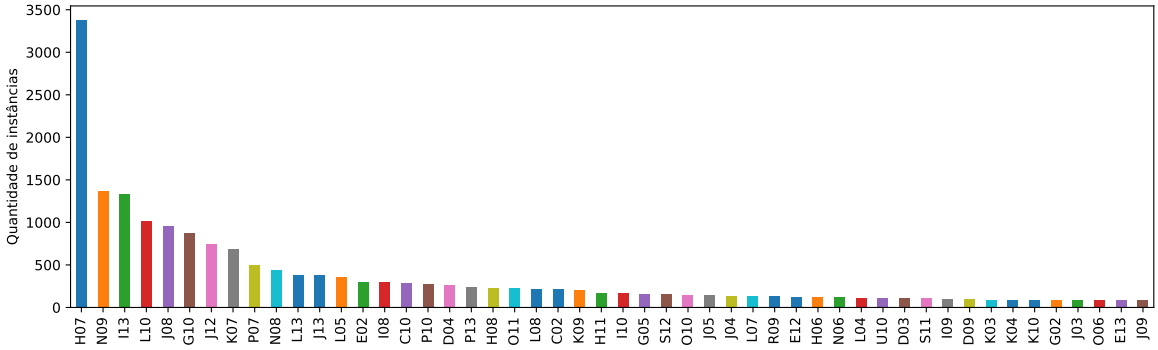
Portanto, neste trabalho são propostas três formas de avaliação de modelos que visam reduzir o viés do conjunto de teste: avaliação para pontos de captura não incluídos no treinamento, avaliação para imagens obtidas posteriormente às utilizadas no treinamento, e avaliação de classes não previamente presentes em determinados pontos de captura no treinamento. As próximas seções deste capítulo avaliam experimentalmente essas condições, mostrando o impacto na avaliação do modelo e, ao final, apresentam recomendações de acordo com os resultados obtidos no estudo de caso da base Snapshot Serengeti.

4.2 Avaliação de modelos para pontos de captura não incluídos no treinamento

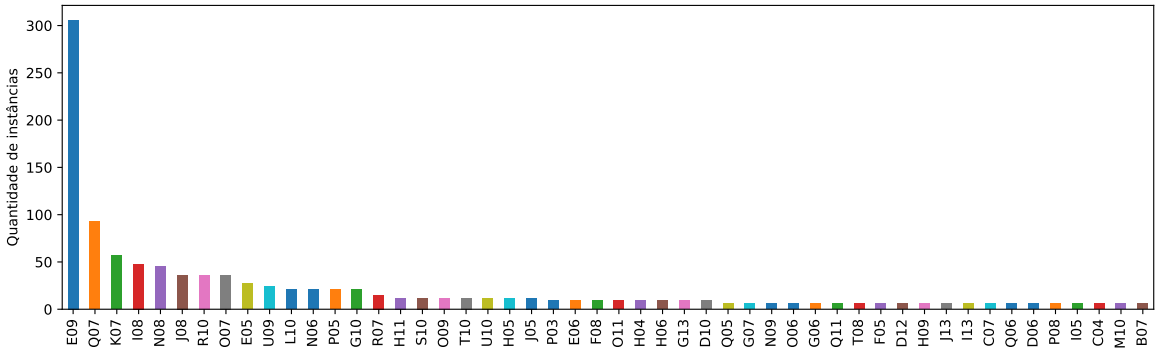
Nesta seção é proposta uma abordagem para avaliação de modelos de identificação de animais, considerando-se as particularidades das bases de imagens de projetos de armadilhas fotográficas em relação à aplicação dos modelos treinados em imagens de



(a) Wildebeest



(b) Guinea Fowl



(c) Secretary Bird

Figura 4.5: Distribuição das espécies (a) Wildebeest, (b) Guinea Fowl e (c) Secretary Bird ao longo dos pontos de captura da base S26.

Fonte: Elaborado pelo autor.

pontos de captura não incluídos no treinamento. Inicialmente, é apresentada uma estratégia de particionamento baseada em pontos de captura. Logo depois, são descritos os experimentos de validação dessa estratégia comparando-a com o particionamento realizado por evento de captura. Em seguida, são apresentados os resultados e, por fim, é realizada uma discussão sobre a eficácia e situações em que a estratégia proposta é recomendada.

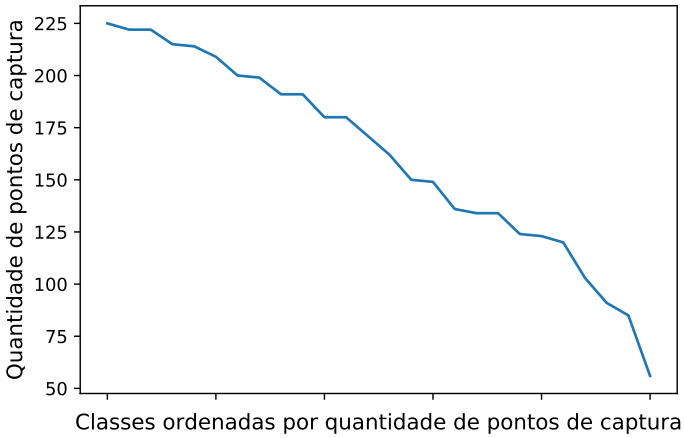


Figura 4.6: Distribuição das classes pelos pontos de captura na base S26.

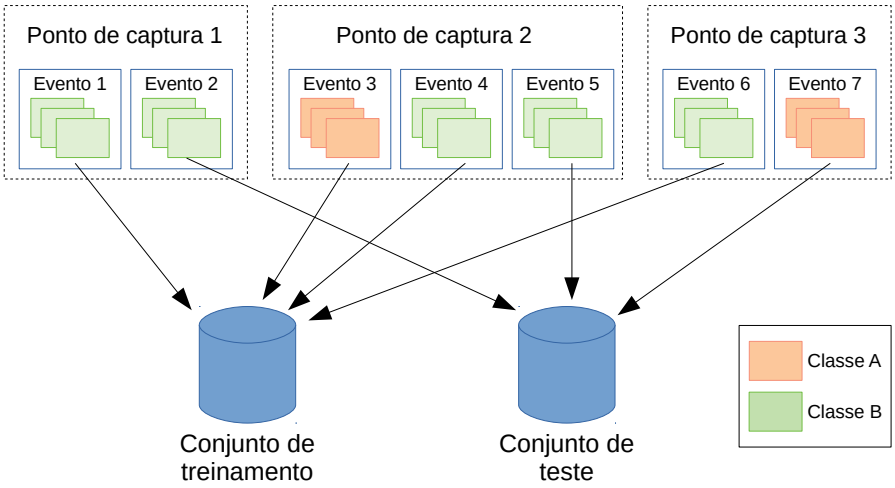
Fonte: Elaborado pelo autor.

4.2.1 Particionamento por ponto de captura

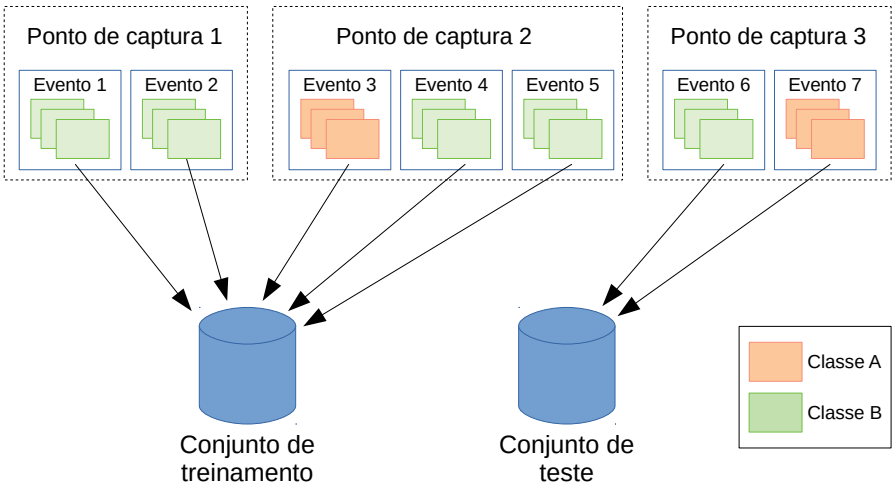
Em um particionamento por ponto de captura todas as imagens de um mesmo local devem ser mantidas juntas na mesma partição (ver Figura 4.7). Mas, diferentemente do particionamento por evento de captura, onde se assume que todas as imagens do conjunto possuem o mesmo rótulo, cada ponto de captura possui imagens de várias classes e, além disso, conforme analisado na Seção 4.1, a distribuição das classes varia de ponto para ponto. Assim, a simples divisão aleatória dos pontos de captura entre as partições pode levar a um sério desbalanceamento entre a quantidade de instâncias no treino e no teste para cada classe. Da mesma forma, pode haver pouca diversidade de locais para algumas classes em alguma das partições. Caso isso ocorra na base de treinamento, pode implicar em um modelo mais enviesado ainda, e caso ocorra na base de teste, pode comprometer a avaliação, devido à baixa variabilidade das instâncias de teste.

Portanto, além de manter as imagens de um mesmo ponto de captura na mesma partição, também é desejável que as partições mantenham a mesma estratificação de classes da base, ao mesmo tempo que tenha certa diversidade de locais para cada espécie. Entretanto, achar uma combinação de pontos de captura que represente a porcentagem desejada para a partição e mantenha essas condições é um problema de combinatória com caráter exponencial. Em razão disso, optou-se por utilizar um algoritmo genético para obter uma combinação que se aproxime desses critérios.

Foi utilizado um algoritmo genético comum, executado por 100 gerações, onde cada geração era composta por 100 indivíduos. Cada indivíduo representa um particionamento possível, indicando em que partição cada ponto de captura deve ser alocado.



(a) Particionamento por evento de captura



(b) Particionamento por ponto de captura

Figura 4.7: Esquema exemplificando o particionamento agrupando-se as imagens por (a) evento de captura e por (b) ponto de captura.

Fonte: Elaborado pelo autor.

A função-objetivo utilizada foi o erro quadrático médio em relação à porcentagem desejada para as partições. Com essa medida de avaliação buscou-se manter a estratificação da base ao penalizar indivíduos que possuíam desbalanceamento muito grande para qualquer uma das classes, ao mesmo tempo que tolera pequenas variações em torno da porcentagem desejada.

4.2.2 Experimentos

Para mensurar o desempenho de modelos quando utilizados para fazer predições em imagens de pontos de captura não incluídos no treinamento, foi realizado um estudo de caso na base S26 por meio de experimentos que comparam os resultados alcançados pelas estratégias de particionamento por evento e por ponto de captura.

Inicialmente a base S26 foi particionada por ponto de captura utilizando a abordagem proposta na seção anterior para separar um conjunto de controle (S26-control), correspondendo a 15% da base, a fim de simular locais não incluídos no treinamento. Como se pode observar na Tabela 4.2, na combinação obtida, todas as classes apresentam uma quantidade de imagens que é muito próxima de 15% do total, mantendo uma estratificação similar à da base inteira. Além disso, a maioria das classes está presente em mais da metade dos pontos de captura da partição de controle, o que garante uma boa diversidade de planos de fundo. Esse conjunto de controle foi utilizado como base de teste para comparar as estratégias de particionamento.

Em seguida, o conjunto de imagens restante (S26-main) foi particionado em treino (70% da S26) e teste (15% da S26) de duas formas: por evento de captura (S26E-train e S26E-test) e por ponto de captura (S26CP-train e S26CP-test). A Figura 4.8 apresenta um esquema resumido desses subconjuntos da base S26.

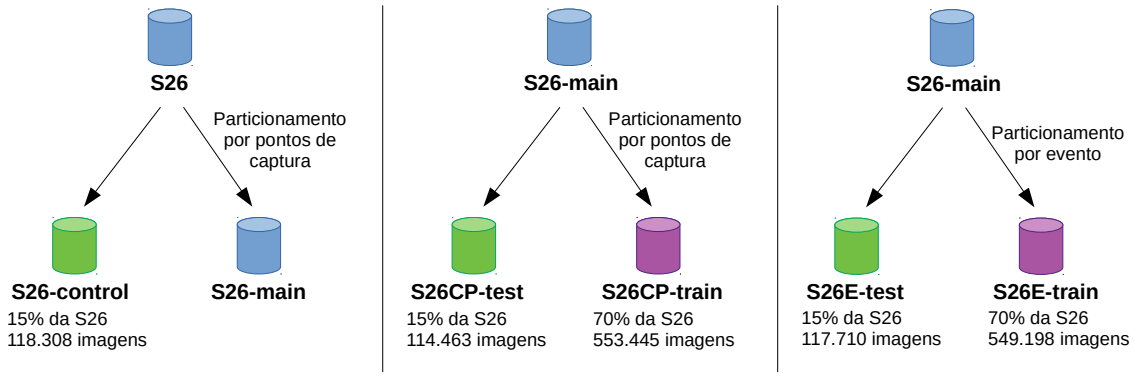


Figura 4.8: Esquema do particionamento da base S26 para os experimentos para avaliação de pontos de captura não incluídos no treinamento.

Fonte: Elaborado pelo autor.

Foram utilizadas as arquiteturas GoogLeNet, InceptionV3 e ResNet18. Cada uma dessas arquiteturas foi treinada separadadamente em ambos os conjuntos de treinamento (S26E-train e S26CP-train) e de duas formas: treinamento completo com inicialização aleatória dos pesos, conforme realizado por Norouzzadeh et al. [2018] e transferência de aprendizado com *fine tuning* da arquitetura, utilizando um modelo

Tabela 4.2: Quantidade de imagens por classe da partição S26-control.

Classe	# Instâncias	% do total da classe	Pontos de captura
wildebeest	36866	15,15%	40
zebra	22867	15,49%	40
gazelleThomsons	16711	14,89%	40
hartebeest	4935	14,58%	39
buffalo	4593	13,86%	34
human	4163	15,86%	40
elephant	3750	14,94%	34
giraffe	3161	14,54%	31
guineaFowl	3154	14,61%	28
impala	3234	14,98%	27
warthog	3315	15,95%	36
gazelleGrants	2958	14,61%	39
hyenaSpotted	1478	14,61%	38
lionFemale	1215	14,48%	31
eland	1056	15,80%	22
topi	824	14,19%	24
baboon	689	15,60%	15
reedbuck	655	15,94%	32
cheetah	480	14,41%	20
dikDik	498	14,98%	22
hippopotamus	480	14,87%	10
lionMale	320	14,89%	21
koriBustard	260	13,94%	17
ostrich	283	15,13%	25
secretaryBird	198	16,38%	14
jackal	165	14,30%	25

pré-treinado na ImageNet. A GoogLeNet e a ResNet18 foram escolhidas para fins de comparação com outros trabalhos que tratam do problema de classificação de animais em imagens de armadilhas fotográficas (Norouzzadeh et al. [2018], Villa et al. [2017], Willi et al. [2018]). Optou-se por acrescentar a InceptionV3 para fins de comparação com uma arquitetura mais robusta em relação ao combate ao *overfitting*.

4.2.2.1 Detalhes da implementação

Como é comum na literatura de aprendizado profundo, durante o treinamento foi utilizado aumento de dados nos conjuntos de treinamento, sendo realizadas as operações de espalhamento horizontal e recorte aleatório. No processo de recorte as imagens tam-

bém foram redimensionadas de acordo com o tamanho de entrada de cada arquitetura. Para a GoogLeNet e ResNet18, as imagens foram redimensionadas para 256x256 *pixels* e realizados recortes aleatórios de 224x224 *pixels*. Para a InceptionV3, os recortes tiveram tamanho de 299x299 *pixels* em imagens redimensionadas para 345x345. Ainda na etapa de pré-processamento, subtrai-se a média das imagens – obtida por meio das partições de treinamento – e o valor dos *pixels* é escalado entre -1 e 1.

Para o treinamento com a GoogLeNet, foi utilizada uma versão ligeiramente modificada, que inclui *batch normalization*, mas mantém o *dropout* de 40% após a camada de *average pooling* ao final da arquitetura. Além disso, os classificadores auxiliares que compunham originalmente as arquiteturas GoogLeNet e InceptionV3 não foram utilizados.

O treinamento foi realizado utilizando-se o algoritmo de otimização Adam com os hiper-parâmetros padrões. Devido à limitação de memória da GPU utilizada, o tamanho do *batch* foi de 128 imagens para a GoogLeNet e a ResNet18, e 32 imagens para a InceptionV3. Para os modelos treinados do zero, o treinamento foi realizado por 55 épocas, de modo que em cada época todas as imagens do conjunto de treinamento são apresentadas ao modelo.

No procedimento de *fine tuning*, utilizou-se como ponto de partida os modelos pré-treinados na ImageNet, substituindo-se a última camada da rede para conter as classes da base S26. Inicialmente, os pesos de todas as camadas foram congelados, com exceção da última camada que foi treinada por cinco épocas com a taxa de aprendizado padrão do Adam de 0,001. Em seguida, ao adicionar novas camadas para o treinamento, essa taxa foi reduzida para 0,0001 para evitar que se modificasse em demasia os descritores já aprendidos com a ImageNet. Um procedimento comum de *fine tuning* consiste em adicionar camada após camada para ser treinada. No entanto, devido o *design* das arquiteturas Inception contar com várias camadas em "paralelo", optou-se por acrescentar camadas de acordo com esses módulos. Com o objetivo de manter um mesmo padrão de experimentos, as camadas da ResNet18 também foram adicionadas ao treinamento em blocos, utilizando-se como referência o tamanho das saídas das camadas em comparação com a GoogLeNet. A Tabela 4.3 apresenta um resumo de quais camadas foram treinadas ao longo das épocas.

4.2.3 Resultados e discussão

Os resultados estão organizados da seguinte maneira. Inicialmente é apresentada uma comparação dos modelos treinados no subconjunto S26E-train em relação aos trabalhos da literatura. Em seguida, são apresentados os resultados dos modelos quando avaliados

Tabela 4.3: Configurações de *fine tuning* das redes para a base S26.

Épocas	Taxa de aprendizado	Camadas treinadas		
		GoogLeNet	InceptionV3	ResNet18
1-5	0,001	Somente última camada	Somente última camada	Somente última camada
6-8	0,0001	Módulo inception (4e) em diante	Módulo inception Mixed_7a em diante	Camada conv5_1 em diante
9-11	0,0001	Módulo inception (4a) em diante	Módulo inception Mixed_6a em diante	Camada conv4_1 em diante

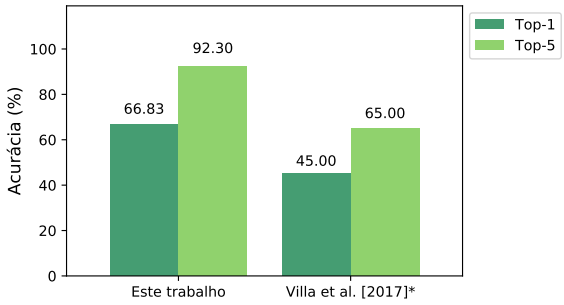
no subconjunto S26-control. Por fim, são apresentados os resultados na avaliação classe a classe.

4.2.3.1 Acurácia dos modelos treinados na S26E-train em comparação com a literatura

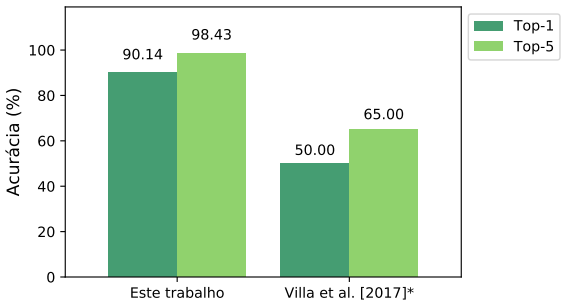
A Figura 4.9 apresenta os gráficos das acurácias top-1 e top-5 dos modelos GoogLeNet e ResNet18, quando treinados no conjunto S26E-train e avaliados no conjunto S26E-test, isto é, utilizando a abordagem de particionamento por evento, e compara com os trabalhos de Villa et al. [2017] e Norouzzadeh et al. [2018].

Comparando-se com os resultados de Villa et al. [2017] para a arquitetura GoogLeNet, obteve-se resultados superiores tanto para simples transferência de aprendizado (modelo ao final da época 5 do procedimento de *fine tuning* adotado) – com uma acurácia top-1 em torno de 66,8% contra 45% – quanto para o *fine tuning* – 90,1% contra cerca de 50%. Apesar de utilizar aproximadamente a mesma quantidade de imagens para treinamento que este trabalho (550 mil), Villa et al. [2017] não utilizam aumento de dados na base, o que pode facilmente explicar essa ampla diferença no desempenho.

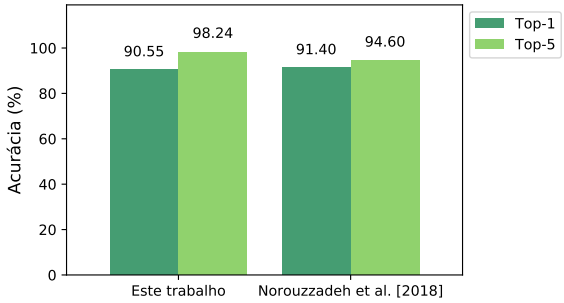
Em contrapartida, os modelos treinados do zero obtêm resultados ligeiramente inferiores aos de Norouzzadeh et al. [2018]. Isso pode ser justificado pelo fato do presente trabalho utilizar uma porcentagem menor da base para o treinamento: 70% (cerca de 550 mil imagens) contra 93% da base (por volta de 730 mil imagens) no trabalho de Norouzzadeh et al. [2018]. Assim, pode-se assumir que os modelos treinados neste trabalho possuem desempenho compatível com a literatura quando avaliados em um conjunto de teste particionado por evento de captura.



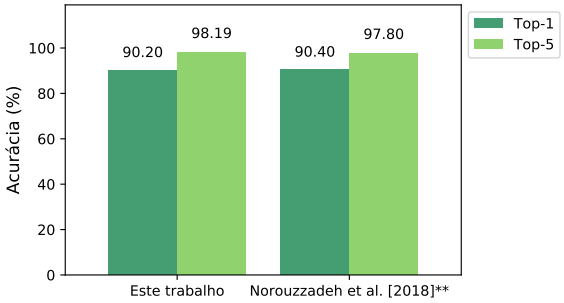
(a) GoogLeNet com treinamento só da última camada.



(b) GoogLeNet com *fine tuning*.



(c) GoogLeNet treinada do zero.



(d) ResNet18 treinada do zero.

Figura 4.9: Gráficos das acurácias top-1 e top-5 dos modelos treinados na S26E-train em comparação com outros trabalhos na literatura.

Fonte: Elaborado pelo autor.

* Estimado a partir dos gráficos apresentados no trabalho de Villa et al. [2017].
** Norouzzadeh et al. [2018] utilizaram as demais classes minoritárias da base Snapshot Serengeti para o treinamento da ResNet18.

4.2.3.2 Desempenho dos modelos para pontos de captura não incluídos no treinamento

O desempenho de todos os modelos, independentemente da arquitetura ou forma de treinamento, diminuiu significativamente quando foram utilizados para identificar animais em imagens do conjunto de controle (S26-control) composto por pontos de captura não incluídos no treinamento. Como mostra a Figura 4.10, essa queda variou entre 6% para a InceptionV3 a mais de 10% para a ResNet18.

Na Figura 4.10, também é possível observar que o desempenho das arquiteturas quando treinadas do zero é muito similar no conjunto particionado por evento (S26E-test). No entanto, no conjunto de controle (S26-control) há uma variação bem maior, com quase 4% de diferença a mais entre a acurácia da InceptionV3 e da ResNet18 quando treinadas do zero; a diferença ultrapassa os 5% quando comparada à InceptionV3 treinada com *fine tuning*. Esses resultados enfatizam a necessidade de uma

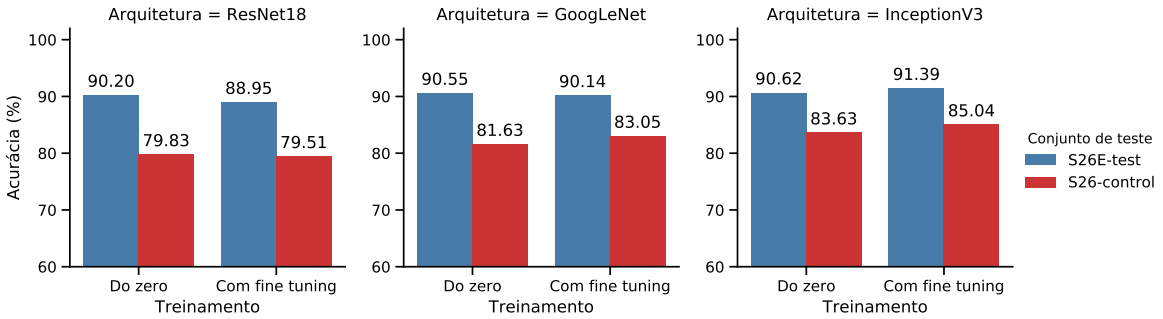


Figura 4.10: Gráficos da acurácia top-1 dos modelos treinados no conjunto S26E-train.

Fonte: Elaborado pelo autor.

avaliação cuidadosa dos modelos caso seja necessário reconhecer animais em imagens de pontos de captura não incluídos no treinamento, tais como as obtidas em uma expansão dos projetos para mais locais.

Na Figura 4.11 são exibidos os gráficos do F1-score das classes para os modelos treinados na S26E-train. As classes foram agrupadas em bins de acordo com a quantidade de instâncias de treinamento. Pode-se observar a queda do desempenho em função da diminuição do número de imagens de treinamento, comportamento já identificado na literatura [Horn et al., 2018, Willi et al., 2018]. No entanto, essa redução é consideravelmente maior quando o modelo é avaliado no conjunto de controle, principalmente para as classes minoritárias (com 500 a 10 mil instâncias de treinamento). Nesse caso, se as classes minoritárias forem exatamente as classes de interesse do projeto de monitoramento, uma avaliação por evento de captura pode superestimar a capacidade do modelo em identificar essas classes em novos pontos de captura.

Os modelos treinados na S26CP-train obtiveram resultados similares quando se compara o desempenho no conjunto S26CP-test e no conjunto S26-control, conforme mostra a Figura 4.12. Nesse gráfico também fica evidente o desempenho superior da InceptionV3 em ambos os conjuntos de teste, o que é esperado, uma vez que este modelo tem capacidade maior do que os demais.

Em relação ao particionamento por evento, o particionamento por ponto de captura reduz a variedade de planos de fundo vistos pelo modelo durante o treinamento, uma vez que separa determinados locais para o conjunto de validação. No entanto, a Figura 4.13, que compara os modelos treinados na S26E-train e na S26CP-train, mostra que o desempenho dos modelos no conjunto de controle variou muito pouco, independente do conjunto utilizado para treinamento de cada modelo. O mesmo resultado se repete ao analisar-se os gráficos do F1-score das classes na Figura 4.14.

Portanto, pode-se concluir que o particionamento por ponto de captura proposto

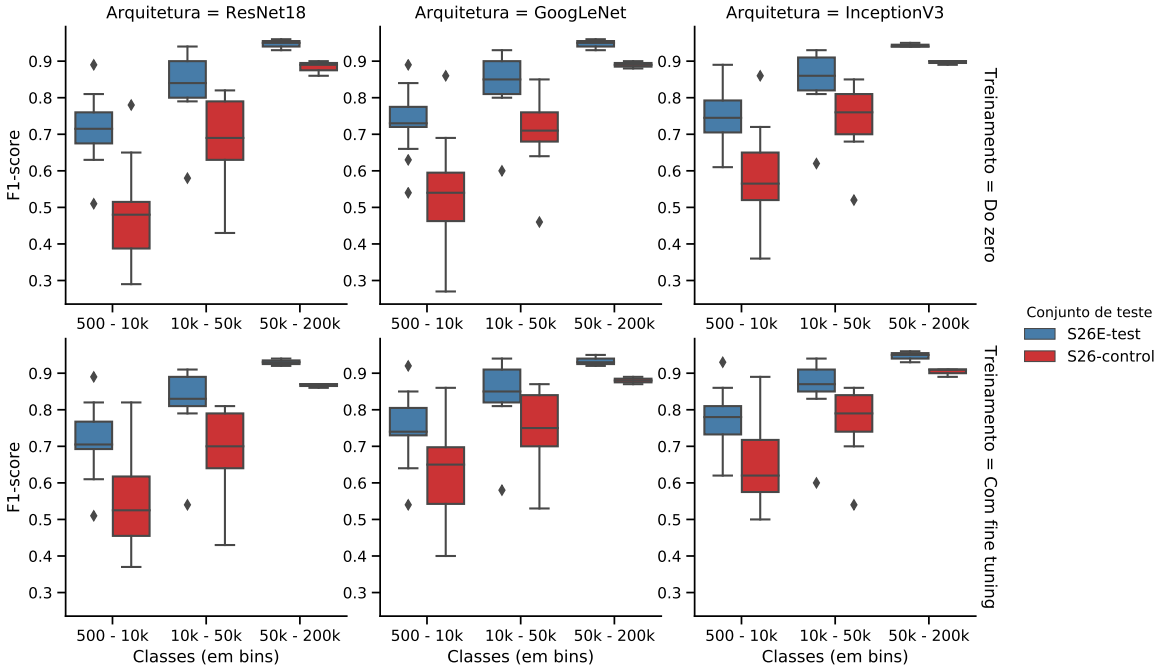


Figura 4.11: Gráficos do F1-score para os modelos treinados na S26E-train.

Fonte: Elaborado pelo autor.

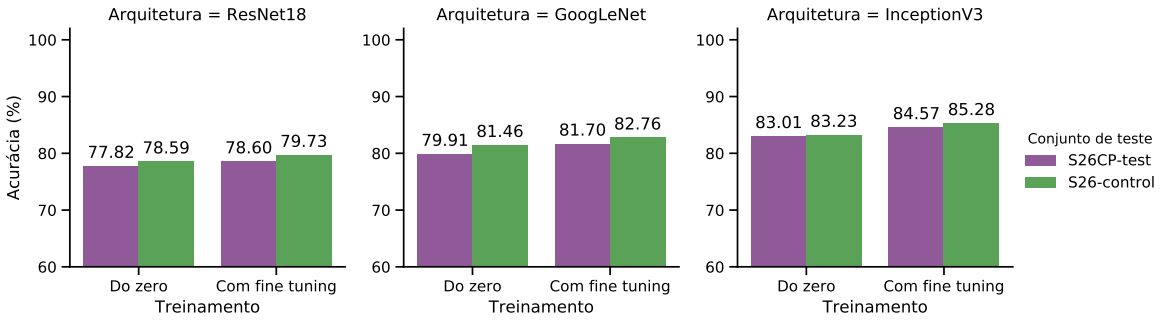


Figura 4.12: Gráfico da acurácia top-1 dos modelos treinados no conjunto S26CP-train.

Fonte: Elaborado pelo autor.

é efetivo na avaliação do desempenho dos modelos para imagens de novos pontos de captura. Dado que os experimentos mostraram que o desempenho dos modelos é muito diferente para pontos de captura não vistos durante o treinamento, essa estratégia de avaliação proposta é recomendada para projetos em que novos pontos de captura podem ser adicionados após o treinamento do modelo. Também é recomendada para novos projetos que irão utilizar modelos treinados em imagens de outras redes de monitoramento. Assim, a avaliação em um conjunto de imagens de validação menos enviesado nos locais de uma rede específica oferece, nos casos citados ou condições

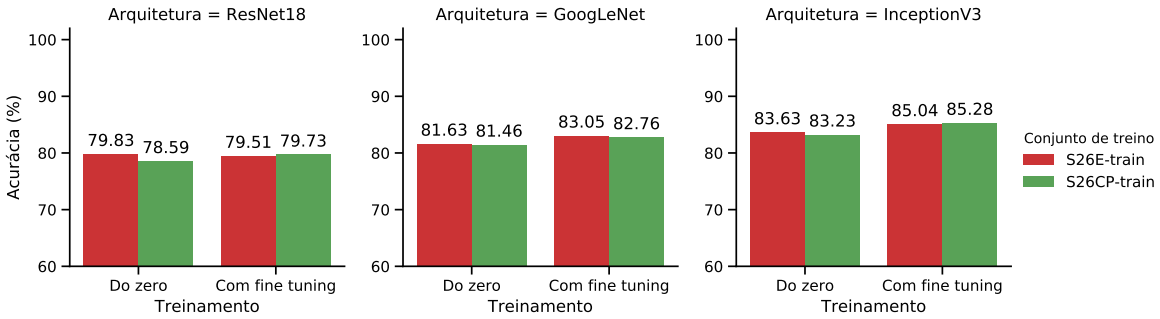


Figura 4.13: Gráfico da acurácia top-1 comparando o desempenho dos modelos treinados nos conjuntos S26E-train e S26CP-train quando avaliados no S26-control.

Fonte: Elaborado pelo autor.

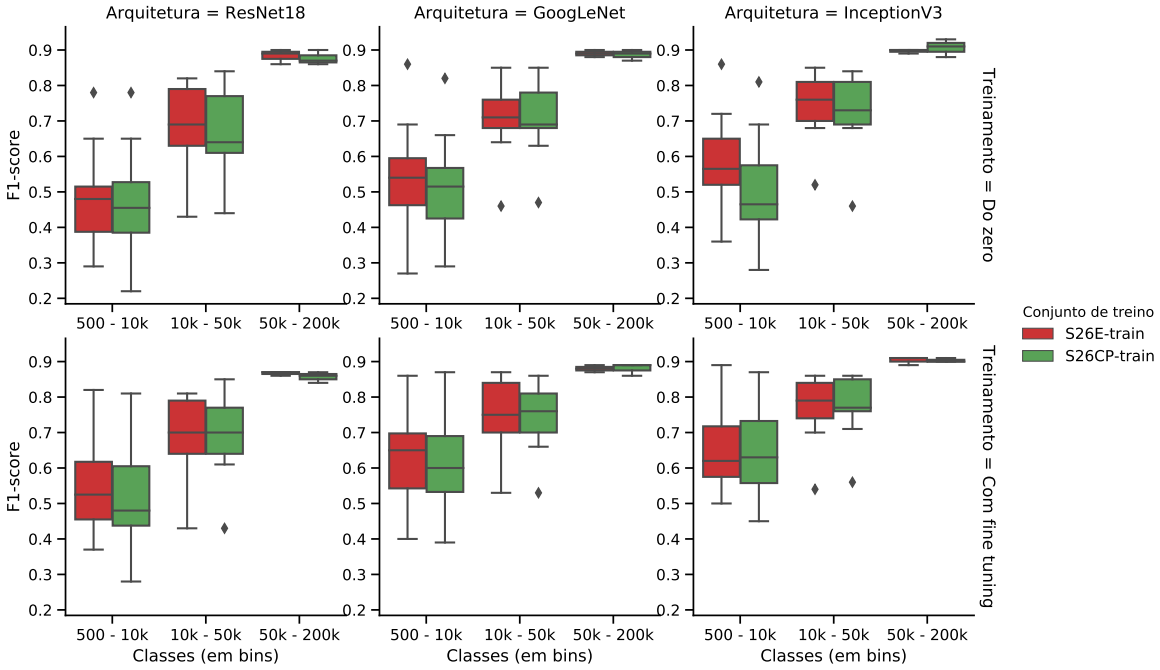


Figura 4.14: Gráficos do F1-score comparando o desempenho dos modelos treinados nos conjuntos S26E-train e S26CP-train quando avaliados no S26-control.

Fonte: Elaborado pelo autor.

similares, indicadores mais realistas para a escolha dos melhores modelos, bem como o ajuste dos parâmetros de treinamento dos mesmos.

4.3 Avaliação de modelos com conjunto de teste com imagens obtidas posteriormente às utilizadas no treinamento

Na seção anterior ficou evidente que a utilização de conjuntos de teste particionados por evento de captura pode superestimar a capacidade dos modelos de identificação de animais em imagens de pontos de captura não incluídos no treinamento. Entretanto, caso o modelo seja utilizado apenas para imagens da rede de câmeras utilizada no treinamento, a hipótese é que o desempenho se mantenha. Mesmo que haja modificações da paisagem ao longo do tempo ou pequenas alterações do ângulo de visão das câmeras, os modelos deveriam ser robustos o suficiente para lidar com isso.

Assim, esta seção irá testar essa hipótese simulando uma situação real de utilização dos modelos treinados: identificar imagens obtidas posteriormente às utilizadas durante o treinamento. Nesse caso, a simulação será feita por meio de um particionamento dos dados em treino e teste com base no tempo: as imagens até uma determinada data ficam no conjunto de treinamento, enquanto as imagens posteriores a essa data ficam no conjunto de teste (ver Figura 4.15).

4.3.1 Experimentos

Para avaliar se o desempenho do modelo se mantém ao longo do tempo se for utilizado apenas para identificar animais em imagens da rede de câmeras utilizada no treinamento, a base S26 foi particionada em treino e teste baseado no período de captura das imagens.

Foi definido que todas as imagens obtidas até o dia 14 de junho de 2012 iriam compor o conjunto de treinamento (S26D-train) e as imagens posteriores a essa data iriam compor o conjunto de teste. Essa data foi escolhida a fim de possibilitar que o conjunto de treinamento deste experimento tenha aproximadamente a mesma quantidade de imagens do experimento com particionamento por evento de captura, ou seja, 550 mil instâncias. Assim, pode-se comparar modelos treinados sob condições similares, apesar da diferença na estratégia de particionamento em treino e teste.

O conjunto de teste corresponde às imagens coletadas a partir de 15 de junho de 2012 até o final do mês de abril de 2013, abrangendo aproximadamente onze meses. Como a partir de fevereiro de 2012 foram adicionados 25 novos pontos de captura no projeto Snapshot Serengeti, o desempenho dos modelos nesses novos locais será inferior, fato comprovado no experimento anterior. Portanto, optou-se por remover todas

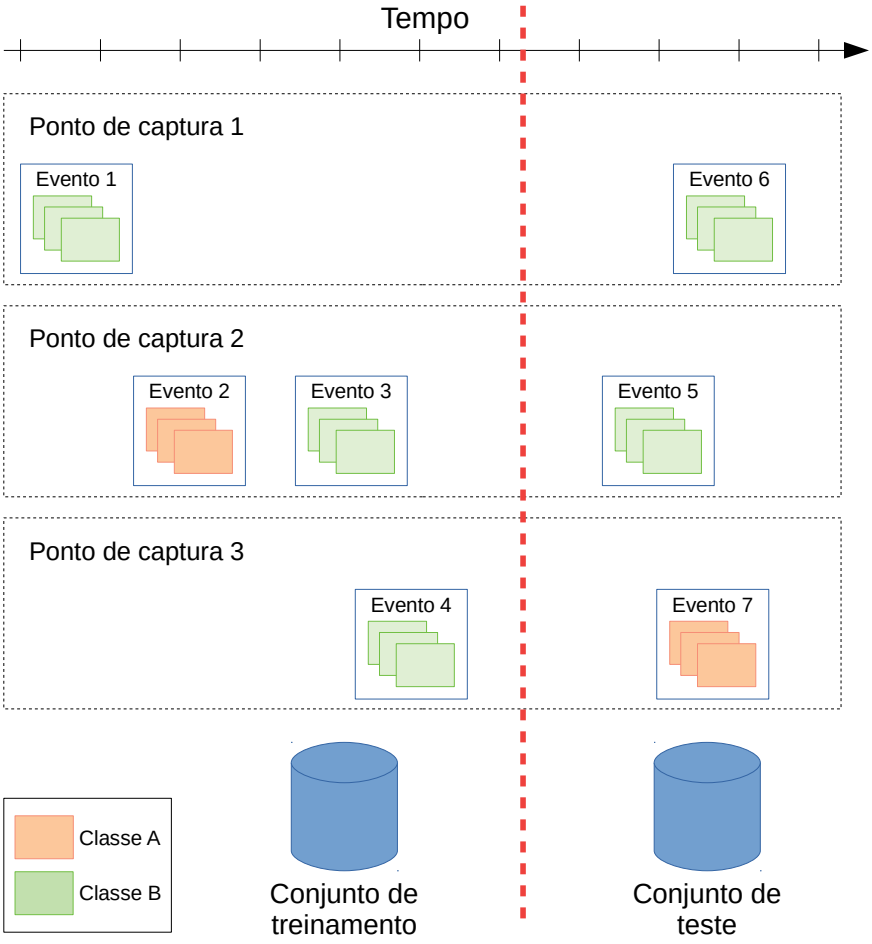


Figura 4.15: Esquema exemplificando o particionamento por tempo de uma base de imagens de armadilhas fotográficas.

Fonte: Elaborado pelo autor.

as imagens dos 25 locais adicionais da base de teste. Dessa forma, nesta série de experimentos, somente os 200 locais originais do projeto (S26D-test200) serão avaliados. No entanto, a pequena quantidade de imagens coletadas entre os meses de fevereiro e junho de 2012 nesses novos pontos de captura foram mantidas no conjunto de treinamento. Além disso, pela forma como o conjunto de teste por tempo foi construído, não é possível garantir que terá a mesma estratificação da base de treinamento, visto que não se pode controlar a frequência com que os animais aparecem. Mas, como os demais experimentos deste capítulo utilizam conjuntos de teste que mantêm a estratificação original dos dados, foi criada uma versão subamostrada da S26D-test200 que mantém a mesma estratificação da base de treino (S26D-test200bal) para fins comparativos. A Figura 4.16 apresenta um esquema representando esse particionamento.

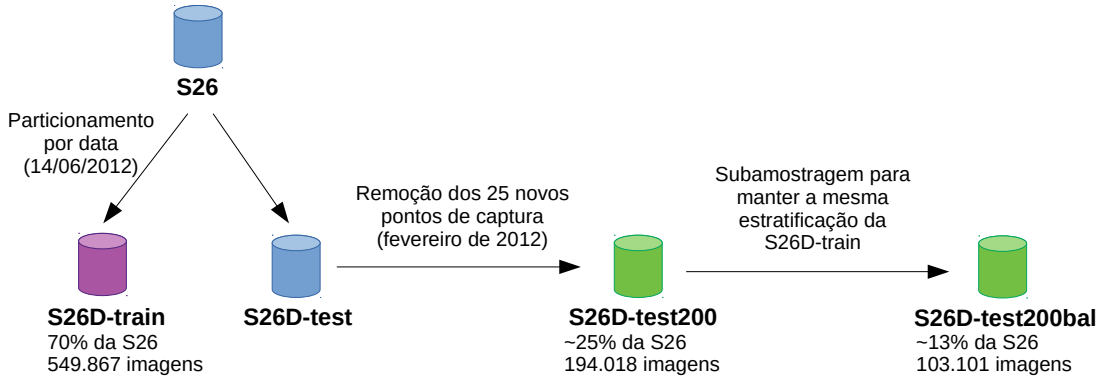


Figura 4.16: Esquema do particionamento da base S26 para os experimentos para avaliação de imagens obtidas posteriormente às utilizadas no treinamento.

Fonte: Elaborado pelo autor.

Para os experimentos desta seção, novamente serão utilizadas as arquiteturas GoogLeNet, InceptionV3 e ResNet18, sendo treinadas de duas formas: treinamento completo com inicialização aleatória dos pesos, e transferência de aprendizado com *fine tuning* das camadas, utilizando modelo pré-treinado na ImageNet. Os procedimentos de pré-processamento das imagens, aumento de dados e demais detalhes de treinamento seguiram o mesmo protocolo adotado e descrito na Seção 4.2.2.1.

4.3.2 Resultados e discussão

Os resultados estão organizados como seguem. Incialmente é feita uma comparação do desempenho dos modelos nas duas versões do conjunto de teste: S26D-test200 e S26D-test200bal. Em seguida, são apresentados os resultados dos modelos na avaliação mês a mês durante o período que compõe o conjunto de teste, a fim de verificar se o desempenho se mantém ao longo do ano. Por fim, os resultados dos modelos treinados com particionamento por tempo são comparados com os treinados nas bases particionadas por evento de captura e por ponto de captura.

4.3.2.1 Desempenho dos modelos no conjunto de teste particionado por tempo e na versão estratificada

Conforme se observa na Figura 4.17, a arquitetura InceptionV3 tem um desempenho superior às demais, com uma acurácia top-1 5% melhor, aproximadamente. Também se percebe que em todas as arquiteturas e formas de treinamento, os modelos apresentam um desempenho cerca de 1% melhor no conjunto de teste balanceado. No entanto, ao

comparar-se o F1-score das classes em ambos os conjuntos de teste, não há diferença significativa entre os resultados (ver Figura 4.18). Assim, as análises seguintes desta seção serão feitas somente tomando o conjunto S26D-test200 como referência.

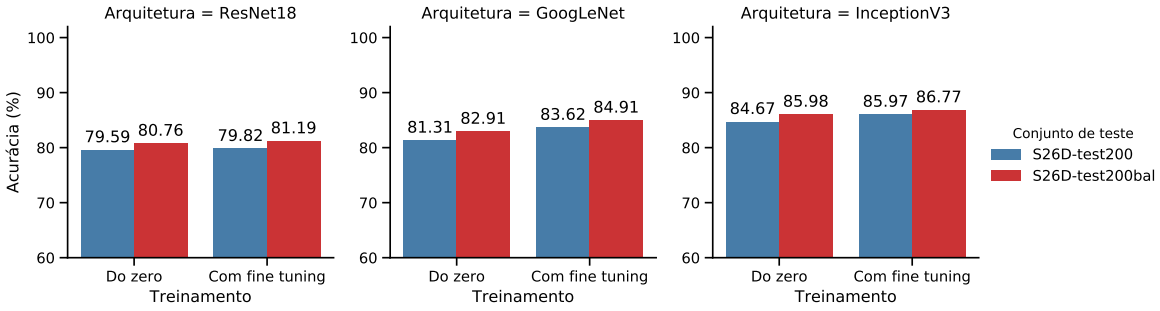


Figura 4.17: Gráficos da acurácia top-1 dos modelos treinados no conjunto S26D-train.

Fonte: Elaborado pelo autor.

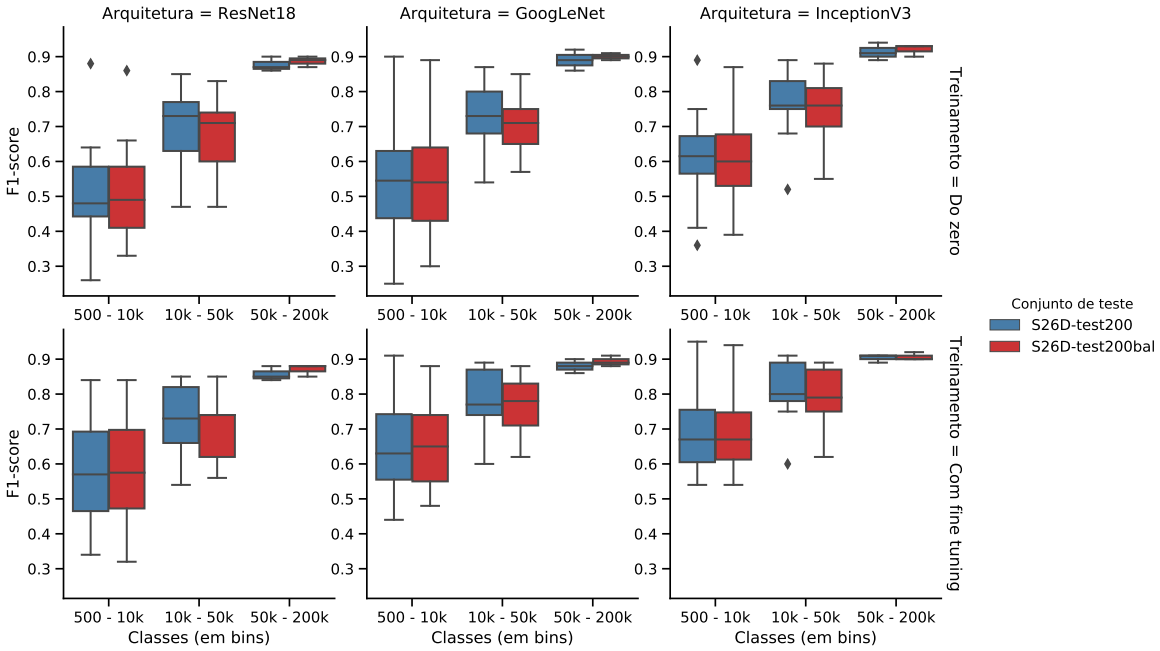


Figura 4.18: Gráficos do F1-score para os modelos treinados na S26D-train.

Fonte: Elaborado pelo autor.

4.3.2.2 Desempenho do modelo treinado na S26D-train ao longo dos meses

A acurácia dos modelos variou bastante de mês para mês. A Figura 4.19 apresenta a acurácia top-1 da arquitetura InceptionV3 em cada mês do conjunto de teste. O bom desempenho para o mês de junho de 2012 de 90,33% era esperado, uma vez que há

imagens no treino obtidas em datas próximas às do teste, e em razão disso, podem ser muito similares. Para os meses seguintes, entretanto, a acurácia decai progressivamente, voltando a ter um novo pico em novembro de 2012. Esse comportamento se deve, principalmente, à variação das observações das classes majoritárias ao longo do ano, em especial a classe wildebeest (gnu), cuja porcentagem das ocorrências ao mês é exibida na Figura 4.20. A variação dessa classe em específico ocorre devido à grande migração que os gnus fazem no Serengeti ao longo do ano.

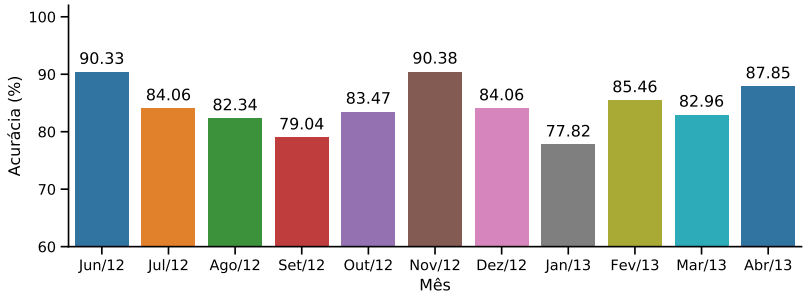


Figura 4.19: Acurácia top-1 ao mês da arquitetura InceptionV3 treinada na S26D-train.

Fonte: Elaborado pelo autor.

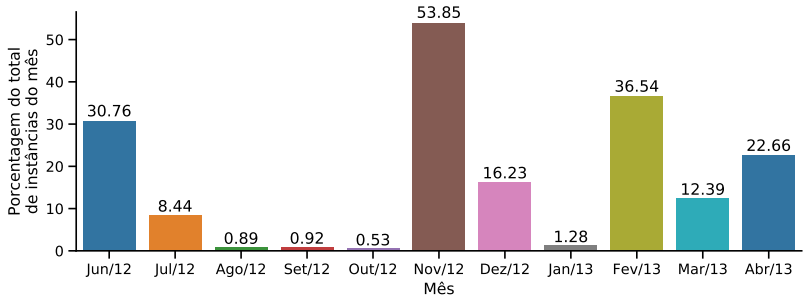


Figura 4.20: Ocorrência da classe wildebeest ao mês no conjunto S26D-test200.

Fonte: Elaborado pelo autor.

Esses resultados indicam que deve-se atentar para a construção do conjunto de teste com base no tempo, de forma que se tenha uma representação das espécies cuja observação é sazonal. Além disso, a avaliação das classes, com métricas como o F1-score, faz-se necessária para a tomada de decisões sobre o desempenho dos modelos.

4.3.2.3 Comparação do desempenho entre particionamento por tempo com as outras formas de particionamento

Os modelos avaliados no conjunto de teste S26D-test obtiveram uma acurácia substancialmente inferior em relação ao alcançado com o particionamento por evento de captura, conforme pode ser visto na Figura 4.21. Esse resultado também pode ser observado ao analisar-se o F1-score para as classes, cujos valores são significativamente inferiores no conjunto de teste particionado por tempo, principalmente para as classes minoritárias, como mostra a Figura 4.22.

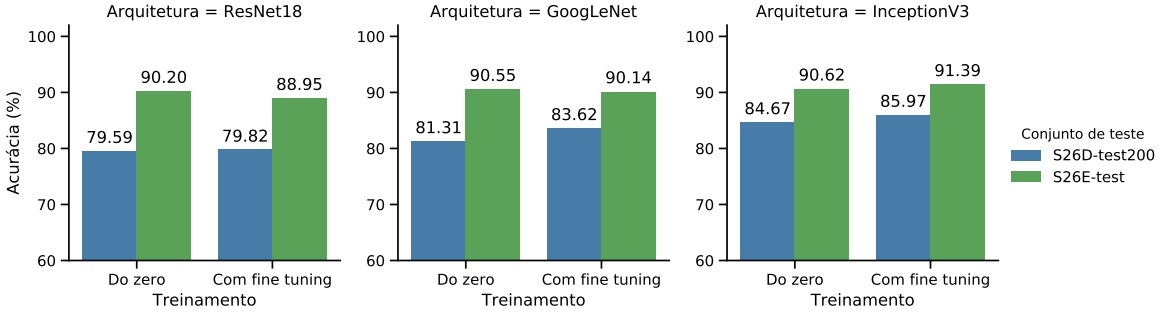


Figura 4.21: Gráficos da acurácia top-1 comparando o desempenho dos modelos com particionamento por tempo e por evento de captura.

Fonte: Elaborado pelo autor.

Também é possível observar uma diferença discrepante entre o desempenho das arquiteturas no conjunto de teste particionado por tempo, diferente da avaliação no conjunto S26E-test, onde o desempenho é similar para todas as arquiteturas e formas de treinamento. Assim, caso os resultados fossem analisados no conjunto S26E-test a fim de escolher um que oferecesse um balanceamento razoável entre acurácia e custo computacional, poderia optar-se por utilizar o modelo ResNet18, que seria menos complexo e teria uma acurácia apenas 0,42% inferior ao da InceptionV3. No entanto, no conjunto S26D-test200, a InceptionV3 apresenta uma acurácia top-1 que ultrapassa em mais de 5% o desempenho da ResNet18. Esse resultado mostra uma capacidade de generalização superior da InceptionV3 para indentificar animais em imagens obtidas posteriormente às utilizadas no treinamento. Assim, pode-se concluir que o particionamento por evento de captura gera um conjunto de teste otimista em relação às situações reais em que um modelo desse tipo teria que lidar mesmo quando utilizado somente em imagens obtidas pelas câmeras da rede utilizada para capturar as imagens de treinamento.

Comparando-se o resultado na base S26D-test200 com o desempenho dos modelos treinados na base S26E-train, mas testados na base de controle S26-control (par-

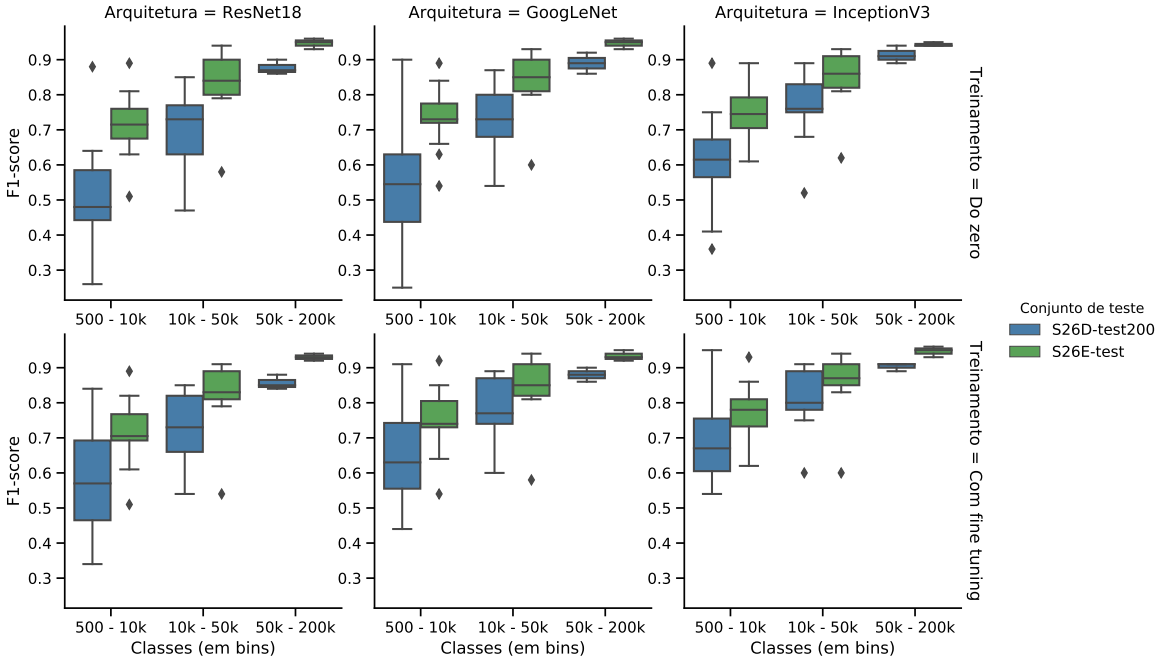


Figura 4.22: Gráficos do F1-score para os modelos treinados e avaliados com particionamento por tempo (S26D) e particionamento por evento de captura (S26E).

Fonte: Elaborado pelo autor.

ticionada por ponto de captura), pode-se perceber um desempenho similar, tanto na acurácia do modelo, conforme mostra a Figura 4.23, quanto ao analisar o F1-score, ilustrado na Figura 4.24. Isso pode indicar que o conjunto de teste particionado por evento não é capaz de evidenciar um possível *overfitting* nos locais de treinamento, que faz com que o desempenho dos modelos decaia com o tempo, à medida que o plano de fundo sofre alterações. Outra hipótese é que os modelos, ao serem treinados em bases de armadilhas fotográficas, aprendem a classificar cenas do animal em determinados planos de fundo, e não o animal em si. Nessa situação, os modelos teriam dificuldade em reconhecer o animal em locais em que essa espécie não foi vista ou foi vista pouquíssimas vezes durante o treinamento. Esta última hipótese será testada na Seção 4.4.

Pode-se concluir que o particionamento por tempo é uma estratégia que gera um conjunto de teste mais realista para avaliar modelos que serão utilizados em projetos em que não se espera a adição de novos pontos de captura. Entretanto, deve-se ter cautela ao selecionar o limiar de separação entre treino e teste, a fim de representar todas as classes, com atenção especial à sazonalidade.

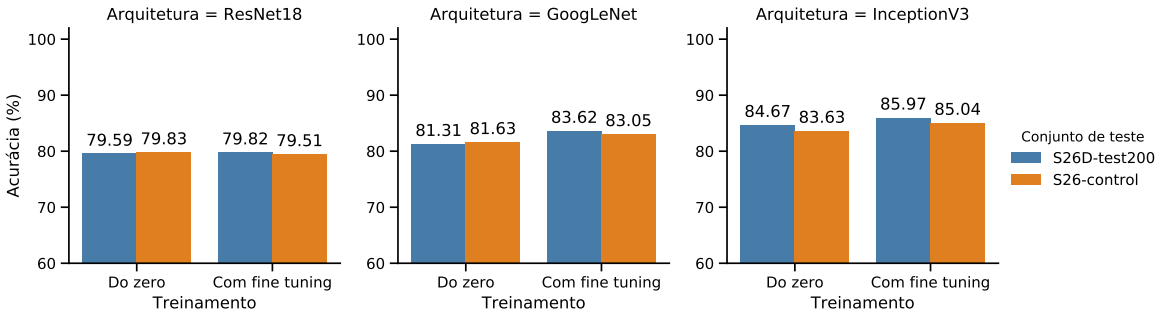


Figura 4.23: Gráficos das acurácia top-1 comparando os resultados S26D-test200 com os modelos treinados na base S26E-train e avaliados na base S26-control.

Fonte: Elaborado pelo autor.

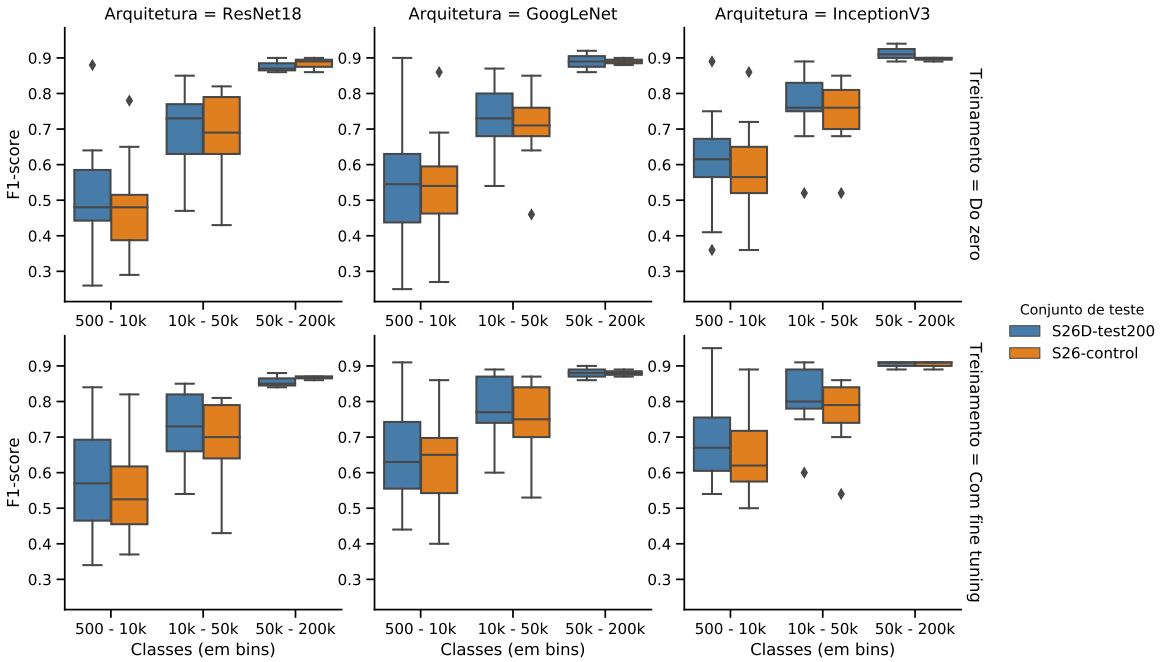


Figura 4.24: Gráficos do F1-score comparando os resultados na S26D-test200 com os modelos treinados na base S26E-train e avaliados na base S26-control.

Fonte: Elaborado pelo autor.

4.4 Avaliação da capacidade de predição de classes não previamente presentes em determinados pontos de captura no treinamento

Nesta seção será avaliada a capacidade dos modelos em identificar classes em pontos de captura nos quais não foram observadas durante o treinamento. Essa condição pode

ocorrer principalmente para classes minoritárias, cujas imagens foram obtidas de uma quantidade menor de pontos de captura.

Para realizar tal avaliação, foi realizado um particionamento dos pontos de captura classe a classe para simular essa condição. Sendo assim, para cada classe, todas as imagens pertencentes ao conjunto de treinamento são oriundas de pontos de captura diferentes de qualquer ponto de captura das imagens utilizadas no conjunto de teste. Isso significa que, durante o teste, o modelo tenta reconhecer um animal em um local nunca visto durante o treino. A Figura 4.25 mostra um exemplo onde para a classe A, as imagens do ponto de captura 2 ficaram no treino e do ponto de captura 3 no teste, enquanto para a classe B, as imagens dos pontos de captura 1 e 3 ficaram no treino e as do 2 no teste. Nessa abordagem, a escolha da partição é independente entre as espécies, com isso pode haver imagens de um mesmo ponto de captura no treino e no teste, no entanto, serão de classes diferentes.

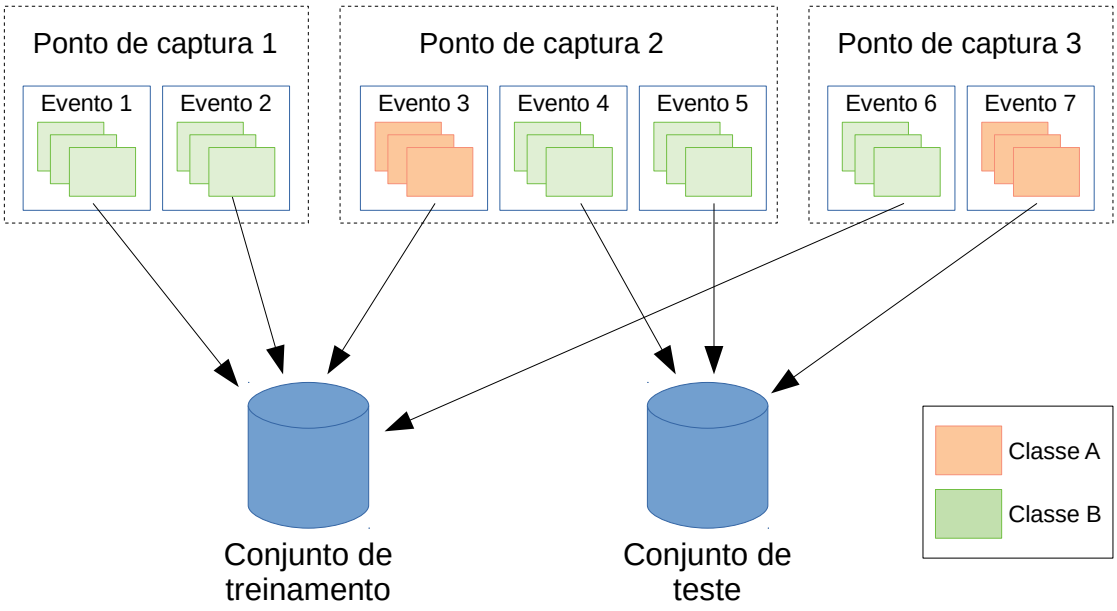


Figura 4.25: Esquema exemplificando particionamento dos pontos de captura por classe.

Fonte: Elaborado pelo autor.

4.4.1 Experimentos

Os experimentos desta seção foram comparados com o desempenho no conjunto de controle S26-control descrito na Seção 4.2.1. Por isso, o particionamento dos pontos

de captura foi aplicado no subconjunto chamado de S26-main. Para cada classe, os pontos de captura foram distribuídos aleatoriamente nos conjuntos de treino e teste, de forma que o conjunto de treinamento (S26C-train) pudesse conter 70% das imagens da S26, e o conjunto de teste (S26C-test) aproximadamente 15% da S26, com a S26-control correspondendo aos 15% restantes. Durante o particionamento foi mantida a estratificação das classes igual à existente na base original. A Figura 4.26 apresenta um esquema que resume os subconjuntos acima descritos.

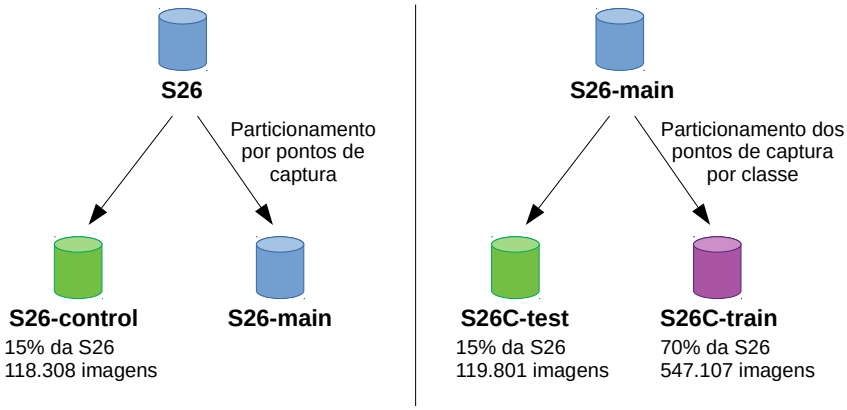


Figura 4.26: Esquema do particionamento da base S26 para a avaliação em relação a classes não previamente presentes em certos pontos de captura no treinamento.

Fonte: Elaborado pelo autor.

Nos experimentos desta seção foram, novamente, utilizadas as arquiteturas GoogLeNet, ResNet18 e InceptionV3, seguindo o mesmo protocolo de treinamento descrito na Seção 4.2.2.1.

4.4.2 Resultados e discussão

A Figura 4.27 mostra que os modelos treinados na S26C-train têm desempenho muito inferior no conjunto de teste S26C-test em comparação com o desempenho no conjunto de controle, com uma diferença de cerca de 10% em todos os modelos. Na avaliação do F1-score das classes, mostrada na Figura 4.28, se repete o desempenho significativamente inferior dos modelos no conjunto de teste S26C-test. Esse resultado mostra que os modelos treinados neste experimento identificam melhor os animais em pontos de captura que não foram incluídos no treinamento do que em locais presentes no treinamento, mas nos quais o animal não foi visto. Isso também sugere que o modelo aprende a identificar cenas e não animais em si, isto é, aprende determinadas espécies em certos planos de fundo.

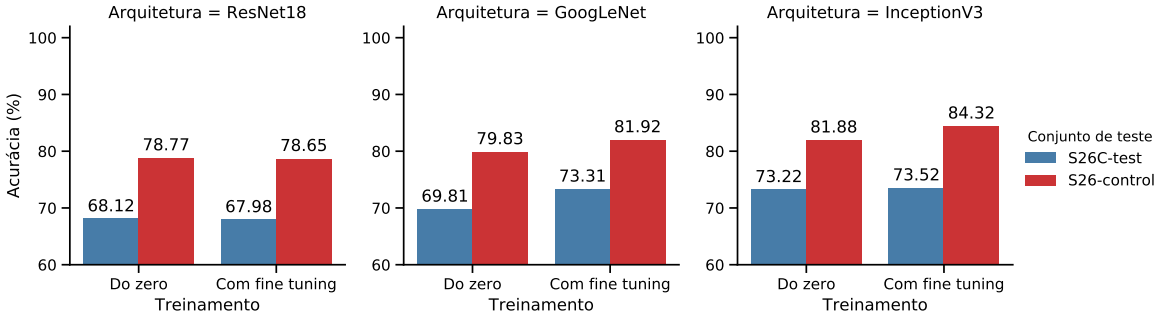


Figura 4.27: Gráficos da acurácia top-1 dos modelos treinados no conjunto S26C-train.

Fonte: Elaborado pelo autor.

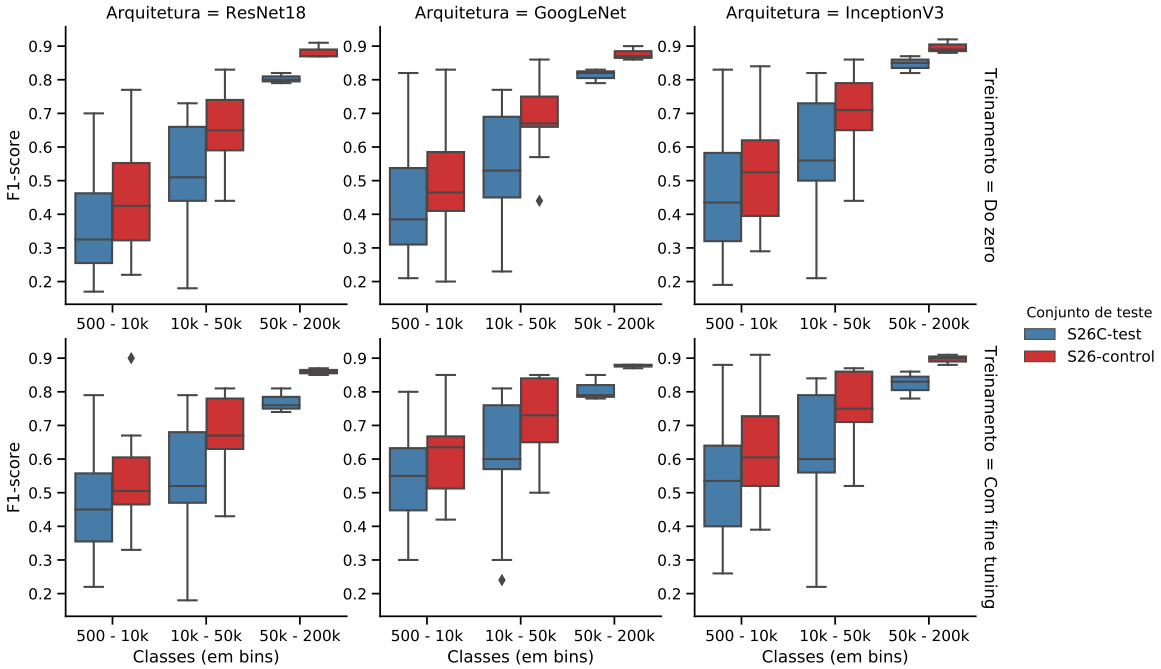


Figura 4.28: Gráficos do F1-score para os modelos treinados na S26C-train.

Fonte: Elaborado pelo autor.

Ao comparar o desempenho dos modelos treinados na S26C-train com os modelos treinados na base S26E-train e testados no conjunto de controle, há uma redução no desempenho, variando de 1% a 2%, dependendo do modelo, como pode ser visualizado na Figura 4.29. No entanto, ao analisar o F1-score na Figura 4.30, pode ser observado que não houve diferenças significativas.

O tipo de particionamento apresentado nesta seção reduz a variedade de classes por ponto de captura vistas durante o treinamento, aumentando o viés dos modelos. Em razão disso, não recomenda-se utilizar essa abordagem para treinamento e avali-

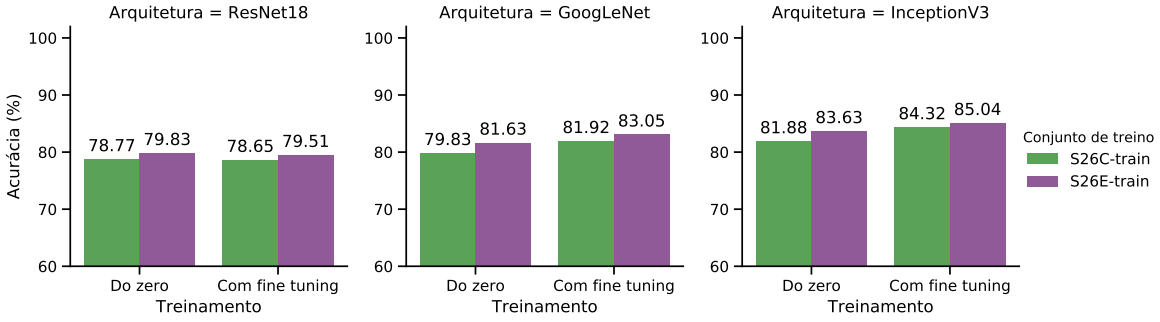


Figura 4.29: Gráficos da acurácia top-1 comparando os resultados dos modelos treinados na S26C-train e S26E-train avaliados no conjunto S26-control.

Fonte: Elaborado pelo autor.

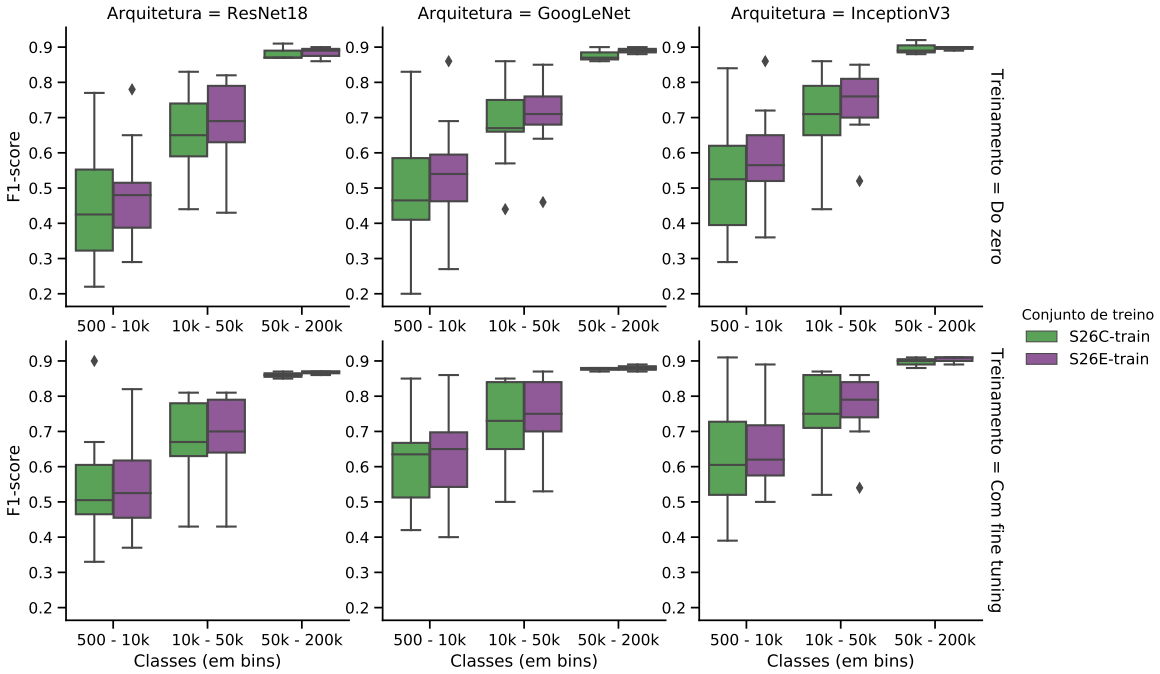


Figura 4.30: Gráficos do F1-score para os modelos treinados na S26C-train e S26E-train e avaliados no conjunto de controle S26-control.

Fonte: Elaborado pelo autor.

ação dos modelos. No entanto, esses experimentos permitiram evidenciar o viés dos modelos em identificar melhor as classes presentes nos pontos de captura utilizados no treinamento. Assim, é desejável que para cada ponto de captura, seja utilizado o maior número possível de classes.

4.5 Considerações finais

Neste capítulo foi realizado um estudo de caso na base S26 a fim de identificar condições de avaliação mais realistas para modelos de identificação de animais em imagens de armadilhas fotográficas. Foram realizados experimentos para validar as abordagem de particionamento propostas para avaliação *out-of-sample*: por ponto de captura, por data e por classes não presentes em determinados pontos de captura. Foi mostrado experimentalmente que a abordagem que utiliza particionamento por evento de captura superestima a capacidade de generalização dos modelos, principalmente para classes minoritárias. Para os casos em que os modelos sejam utilizados para predizer animais em pontos de captura não incluídos no treinamento, como numa expansão dos projetos com adição de novos pontos de monitoramento, a abordagem de particionamento por ponto de captura proposta representa a opção mais realista. Caso os modelos sejam utilizados somente na mesma rede de câmeras utilizada para o treinamento, a estratégia de particionamento por tempo se mostrou mais adequada. No próximo capítulo essas abordagens são aplicadas para avaliar o desempenho de modelos para identificação de animais em projetos de armadilhas fotográficas de menor porte.

5

Avaliação *out-of-sample* da transferência de aprendizado para classificação de animais em imagens de armadilhas fotográficas

Este capítulo apresenta um estudo sobre a transferência de aprendizado para classificação de animais em imagens de armadilhas fotográficas. A Seção 5.1 apresenta as bases de armadilhas fotográficas utilizadas e o particionamento para treinamento e teste adotado. A Seção 5.2 descreve os experimentos realizados e na Seção 5.3 são apresentados os resultados e uma discussão a respeito dos mesmos.

5.1 Bases de dados

Bases de dados em larga escala são essenciais para o sucesso de treinamento de redes neurais profundas. No entanto, em aplicações reais nem sempre há dados rotulados suficientes para realizar o treinamento a partir do zero, como é o caso das bases de armadilhas fotográficas Mamirauá, Caxiuanã e Central Suriname. Nessas situações, utiliza-se transferência de aprendizado com modelos previamente treinados em outras bases.

As próximas subseções apresentam as bases de imagens para as quais foi realizado transferência de aprendizado, bem como descrevem o particionamento em treino, validação e teste que foi realizado utilizando-se as recomendações especificadas no Capítulo 4, de acordo com o protocolo de monitoramento adotado pelo projeto.

5.1.1 Mamirauá

A base fornecida pelo Instituto Mamirauá contém cerca de 127 mil imagens de animais da Floresta Amazônica obtidas através de armadilhas fotográficas instaladas na Reserva de Desenvolvimento Sustentável Mamirauá. Neste trabalho, foi utilizado um subconjunto de 13 espécies, chamado de Mamiraua13, conforme mostra a Tabela 5.1.

Tabela 5.1: Quantidade de imagens por classe da base Mamiraua13.

Classe	Treino	Validação	Teste
Cathartes aura	4483	940	956
Pauxi tuberosa	3692	821	822
Panthera onca	3514	744	758
Tupinambis teguixin	3026	643	662
Crax globulosa	2373	531	537
Homo sapiens	2144	445	456
Leopardus wiedii	1722	364	359
Aramides cajaneus	1583	331	333
Sapajus macrocephalus	1483	330	328
Coragyps atratus	659	153	153
Coendou prehensilis	598	124	128
Saimiri vanzolinii	514	125	108
Nasua nasua	506	111	110
Total	**26297**	**5662**	**5710**

A rotulagem dessa base de imagens foi realizada por especialistas do Instituto Mamirauá, seguindo o padrão da comunidade científica onde o rótulo da classe é dado para o evento de captura como um todo e não para uma imagem em específico. Somando-se a isso o fato de nesse projeto a cada evento de captura obter-se 10 imagens em sequência, muitas imagens rotuladas como sendo de uma classe não contêm nenhum animal na cena. Assim, durante a preparação dessa base as imagens que não continham animal foram removidas manualmente. No entanto, foram mantidas aquelas em que aparecem apenas partes do corpo do animal, conforme procedimento adotado por Villa et al. [2017].

Para os experimentos deste capítulo, a base foi divida em 70% para treinamento (26297 imagens), 15% para validação (5662 imagens) e 15% para teste (5710 imagens). Como, nesse projeto, novos locais podem ser adicionados ao longo do tempo, o divisão em treino e teste foi realizada utilizando-se a abordagem de particionamento por pontos de captura, conforme recomendação do Capítulo 4. Assim como na Seção 4.2.1, também

foi utilizado um algoritmo genético durante o particionamento para garantir a mesma estratificação das classes em todas as partições.

5.1.2 Caxiuanã

A base Caxiuanã [Lima & Santos, 2018] foi fornecida pelo Tropical Ecology Assessment and Monitoring (TEAM) Network. Essa base foi coletada utilizando o protocolo de monitoramento de vertebrados terrestres do projeto TEAM [2011]. Nesse protocolo, os pontos de captura devem ser mantidos ao longo do projeto [1], com as imagens sendo coletadas durante um período de 30 dias seguidos, uma vez por ano, na estação seca. Em razão disso, utilizou-se a abordagem de particionamento por tempo para separar os conjuntos de treinamento (anos de 2010, 2012, 2013 e 2014), validação (ano de 2015) e teste (ano de 2016). Não foi realizada nenhuma verificação manual para remoção de possíveis imagens sem animal na cena. Para este trabalho foram utilizadas 18 espécies que tinham instâncias em todas as partições, conforme detalha a Tabela 5.2.

Tabela 5.2: Quantidade de imagens por classe da base Caxiuana18.

Classe	Treino	Validação	Teste
Dasyprocta leporina	11003	1687	1711
Mazama americana	5889	1930	1303
Pecari tajacu	4753	1194	5178
Mazama nemorivaga	2840	683	1156
Psophia viridis	2712	339	345
Cuniculus paca	2248	495	584
Tapirus terrestris	2160	393	62
Dasypus kappleri	2099	413	159
Dasypus novemcinctus	1430	210	96
Mitu tuberosum	1253	93	192
Nasua nasua	748	112	2
Didelphis marsupialis	746	126	110
Myrmecophaga tridactyla	564	21	42
Puma concolor	494	171	36
Leopardus pardalis	488	51	81
Metachirus nudicaudatus	401	90	43
Panthera onca	279	184	108
Total	**40298**	**8216**	**11231**

[1] Os pontos de captura só devem ser movidos para novas posições em casos de quedas de árvores que obstruam o campo de visão da câmera e não possam ser facilmente removidas.

5.1.3 Central Suriname

Assim como Caxiuanã, a base Central Suriname [Gajapersad, 2018] foi fornecida pelo Tropical Ecology Assessment and Monitoring (TEAM) Network. Como essa base também foi construída seguindo o protocolo de monitoramento de vertebrados terrestres do projeto TEAM [2011], a divisão dos dados foi realizada com a abordagem de particionamento por tempo: treinamento (anos de 2008 a 2014), validação (ano de 2015) e teste (ano de 2016). Pare este trabalho, foram removidas as classes que não tinham instâncias em alguma das partições, resultando num subconjunto composto por 27 espécies que será chamado de CentralSuriname27, conforme mostra Tabela 5.3.

Tabela 5.3: Quantidade de imagens por classe da base CentralSuriname27.

Classe	Treino	Validação	Teste
Dasyprocta leporina	17253	2562	2228
Psophia crepitans	16381	1209	1619
Mazama americana	14509	2503	1069
Crax alector	10901	1184	1658
Myoprocta acouchy	8199	560	798
Pecari tajacu	6252	1208	767
Tapirus terrestris	5743	306	166
Mazama nemorivaga	4942	1046	339
Cuniculus paca	4429	1464	465
Tayassu pecari	4238	2834	1149
Tinamus major	2670	319	233
Dasypus novemcinctus	2121	314	103
Leopardus pardalis	1623	186	149
Didelphis marsupialis	1556	329	191
Dasypus kappleri	1159	94	163
Metachirus nudicaudatus	1065	362	128
Philander opossum	791	119	18
Leopardus wiedii	716	132	69
Leptotila rufaxilla	575	278	50
Puma concolor	563	75	86
Panthera onca	483	90	7
Priodontes maximus	412	33	32
Myrmecophaga tridactyla	386	27	18
Geotrygon montana	381	9	15
Crypturellus variegatus	376	30	29
Odontophorus gujanensis	200	93	51
Eira barbara	193	12	35
Total	**108117**	**17378**	**11635**

5.2 Experimentos

Os experimentos descritos nesta seção foram realizados para verificar quais modelos pré-treinados oferecem melhores condições de transferência de aprendizado para classificação de imagens de armadilhas fotográficas, em especial para bases de menor porte.

Foram utilizadas como origem da transferência de aprendizado as arquiteturas GoogLeNet, ResNet18 e InceptionV3, previamente treinadas na base genérica ImageNet [Russakovsky et al., 2015], na base de armadilhas fotográficas Snapshot Serengeti [Swanson et al., 2015a] e na ImageNet com *fine tuning* na Snapshot Serengeti (ImageNet + Serengeti26). Esses modelos foram treinados para reconhecer animais nas bases Mamiraua13, Caxiuana18 e CentralSuriname27, com duas formas de treinamento: somente da última camada, com o modelo base funcionando como extrator de características, e com *fine tuning*.

O procedimento de treinamento adotado é similar ao utilizado para *fine tuning* no Capítulo 4. Inicialmente, a última camada do modelo utilizado como base do treinamento é alterada para reconhecer as classes da base de destino. Na primeira etapa somente a última camada é treinada utilizando-se o algoritmo de otimização Adam com os hiper-parâmetros padrões, com taxa de aprendizado de 0,001. Nas duas etapas seguintes, mais camadas são adicionadas ao treinamento, conforme especificado na Tabela 5.4, com a taxa de aprendizado sendo reduzida para 0,0001 para evitar que se modificasse em demasia os descritores já aprendidos pelos modelos base.

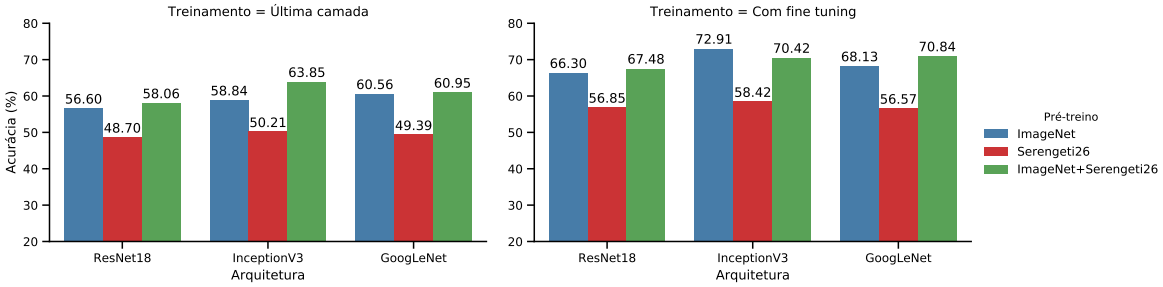
Tabela 5.4: Configurações de *fine tuning* utilizadas para transferência de aprendizado.

Etapa	Taxa de aprendizado	Camadas treinadas		
		GoogLeNet	InceptionV3	ResNet18
1	0,001	Somente última camada	Somente última camada	Somente última camada
2	0,0001	Módulo inception (4e) em diante	Módulo inception Mixed_7a em diante	Camada conv5_1 em diante
3	0,0001	Módulo inception (4a) em diante	Módulo inception Mixed_6a em diante	Camada conv4_1 em diante

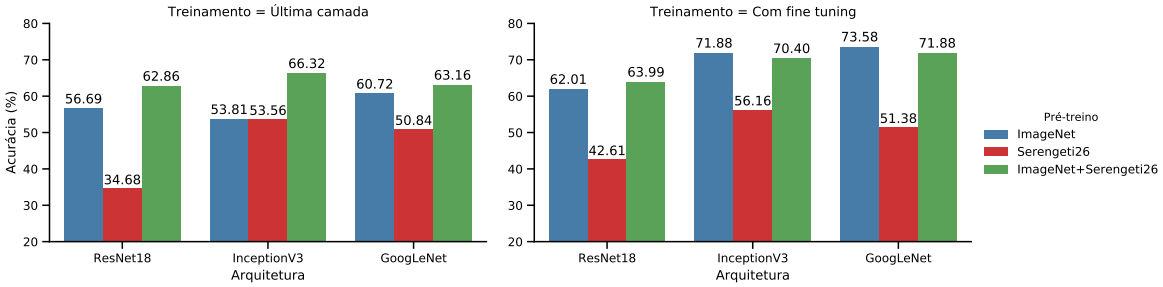
Durante cada etapa dos experimentos deste capítulo, foi utilizada a técnica de parada antecipada do treinamento, utilizando-se como referência a acurácia dos modelos na respectiva base de validação. Assim, caso a acurácia do modelo na base de validação não subisse por mais de 3 épocas, o treinamento era interrompido.

5.3 Resultados e discussões

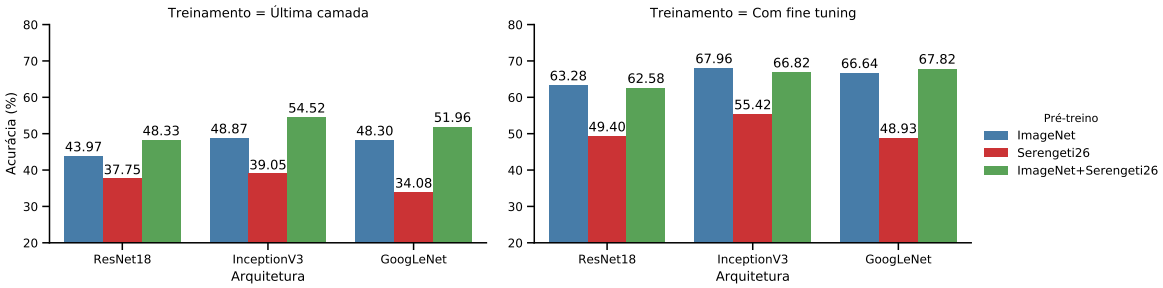
A Figura 5.1 apresenta os resultados da acurácia top-1 para os experimentos relativos à transferência de aprendizado para as bases Mamiraua13, Caxiuana18 e CentralSuriname27. A avaliação do F1-score para as classes é mostrada na Figura 5.2 para a base Mamiraua13, na Figura 5.3 para a base Caxiuana18 e na Figura 5.4 para a base CentralSuriname27.



(a) Acurácia top-1 na base Mamiraua13.



(b) Acurácia top-1 na base Caxiuana18.



(c) Acurácia top-1 na base CentralSuriname27.

Figura 5.1: Gráficos da acurácia top-1 dos modelos treinados nas bases (a) Mamiraua13, (b) Caxiuana18 e (c) CentralSuriname27.

Fonte: Elaborado pelo autor.

No Capítulo 4, os modelos treinados do zero obtiveram resultados similares aos treinados com *fine tuning* na base Serengeti26. Esperava-se que ao transferir apren-

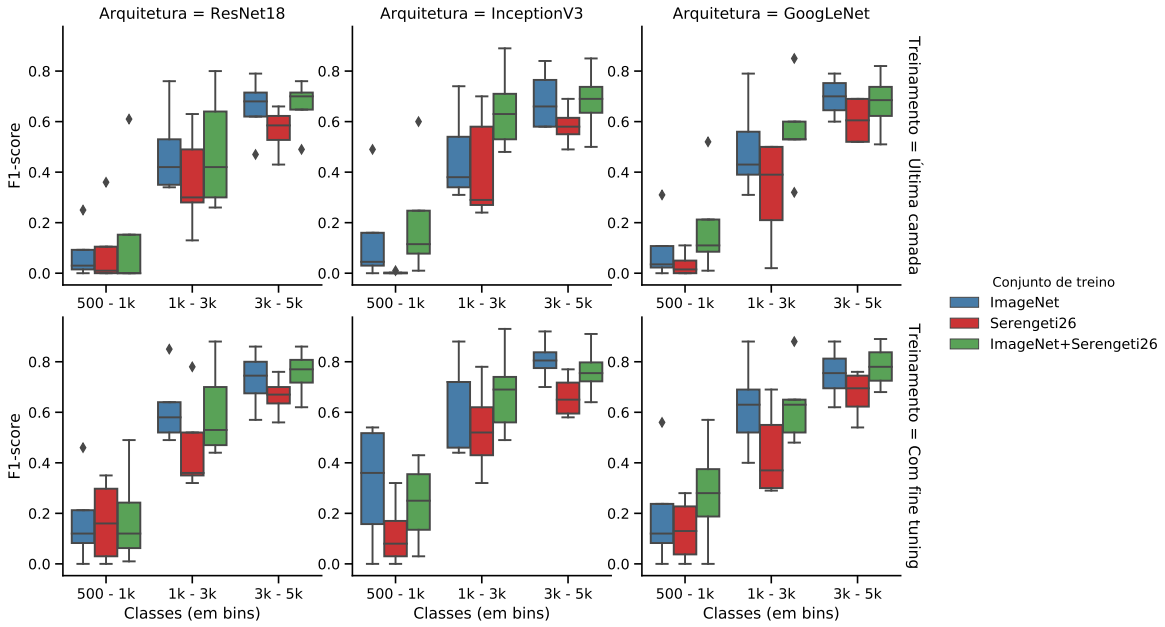


Figura 5.2: Gráficos do F1-score para os modelos treinados na base Mamiraua13.

Fonte: Elaborado pelo autor.

dizado desses modelos para outras bases de armadilhas fotográficas, os resultados fossem similares. Entretanto, como se pode observar, tanto na acurácia geral quanto ao analisar-se o F1-score, os modelos pré-treinados só na base Serengeti26 obtiveram desempenho significativamente inferior aos demais, independente da base de destino, arquitetura ou forma de treinamento. Esse desempenho ruim pode ter acontecido devido à qualidade do treinamento realizado do zero na base Serengeti26. No entanto, o desempenho dos modelos treinados do zero na Serengeti26 foi compatível com a literatura para o particionamento por evento. Portanto, esses resultados reforçam a hipótese do viés otimista do conjunto de teste existente na maioria dos trabalhos da literatura. Além disso, a baixa variabilidade das imagens de armadilhas fotográficas que compõem a base Serengeti26 pode ter prejudicado o aprendizado de características que generalizem bem para outras bases de armadilhas fotográficas.

Analisando-se os resultados para os modelos em que foi treinada somente a última camada, isto é, onde os modelos base funcionam como extratores de características, pode-se perceber que os modelos pré-treinados na ImageNet e com *fine tuning* na Serengeti26 obtiveram resultados superiores aos modelos pré-treinados unicamente na ImageNet, em todas as bases e arquiteturas. Pode-se concluir que essa transferência de aprendizado encadeada (ImageNet+Serengeti26) é vantajosa para bases de armadilhas fotográficas, em especial quando houver poucas instâncias de treinamento e não for

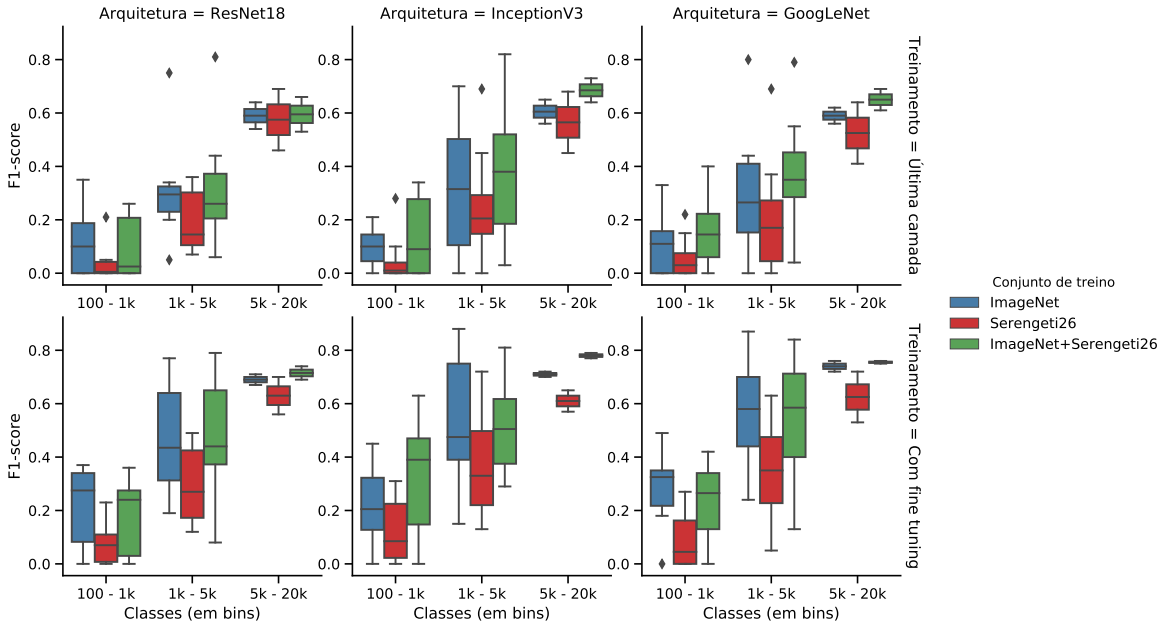


Figura 5.3: Gráficos do F1-score para os modelos treinados na base Caxiuana18.

Fonte: Elaborado pelo autor.

viável fazer *fine tuning* de mais camadas. Entretanto, essa situação de prevalência não se repetiu quando foi realizado *fine tuning* dos modelos. Nesse caso, os experimentos são inconclusivos sobre se é vantajoso fazer transferência de aprendizado encadeada para treinamentos com *fine tuning*.

De maneira geral, dado o custo de treinar modelos do zero e a possibilidade de não se obter um modelo com características genéricas o suficiente, pode-se concluir que seja mais vantajo utilizar transferência de aprendizado através de treinamentos encadeados em outras bases de armadilhas fotográficas, partindo de modelos pré-treinados na base ImageNet.

Os resultados do F1-score em todas as bases reforçam a dificuldade dos modelos em lidar com classes com baixa representatividade. Assim, o desafio da melhoria dos modelos consiste em aprimorar o desempenho para as classes com poucas instâncias de treinamento.

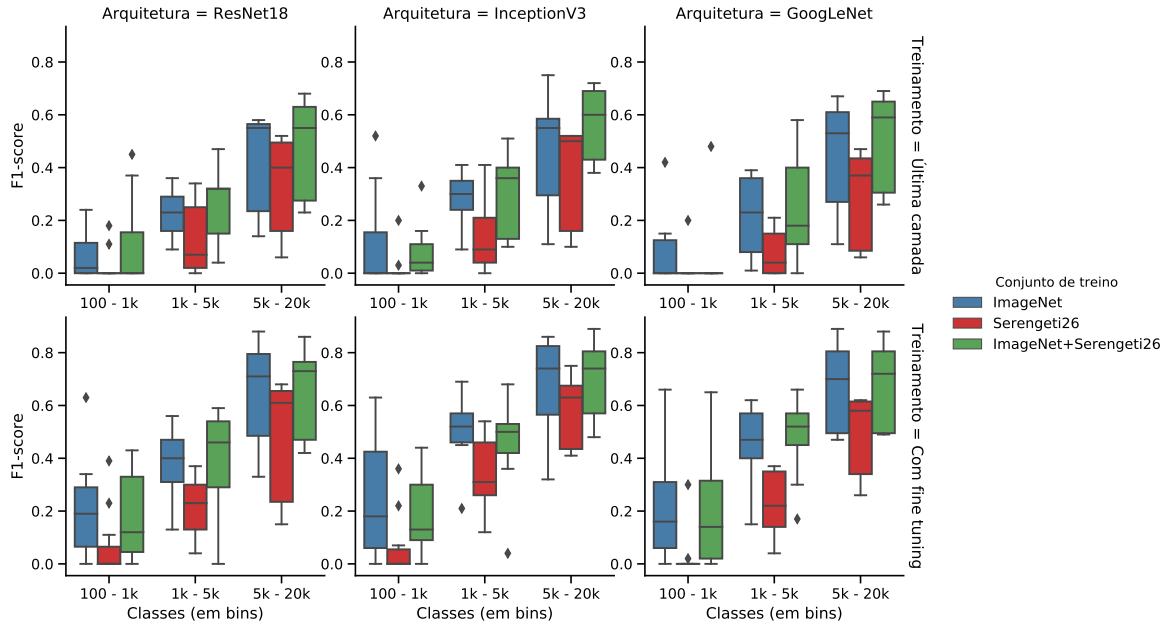


Figura 5.4: Gráficos do F1-score para os modelos treinados na base CentralSuriname27.

Fonte: Elaborado pelo autor.

5.4 Considerações finais

Neste capítulo foi apresentado um estudo de avalição da transferência de aprendizado para bases de imagens de armadilhas fotográficas. Os experimentos realizados mostraram que pode não ser vantajoso transferir aprendizado de modelos treinados do zero em bases de imagens de armadilhas fotográficas, sendo preferível utilizar modelos com transferência de aprendizado de outras bases de armadilhas fotográficas através de treinamento encadeado em modelo pré-treinado na ImageNet.

6

Conclusões

Neste capítulo são apresentadas as conclusões sobre este trabalho, os resultados alcançados e as limitações do estudo. Também são indicadas questões que precisam de maior investigação no contexto de extração automática de informações de imagens de bases de armadilhas fotográficas.

6.1 Considerações finais

Neste trabalho foi realizado um estudo sobre abordagens de particionamento de dados para avaliação *out-of-sample* de modelos de classificação de animais em imagens de armadilhas fotográficas.

A partir de um estudo de caso na base Snapshot Serengeti, foram identificadas situações *out-of-sample* com as quais um modelo pode ter que lidar durante a utilização em condições reais. Foi mostrado experimentalmente que o particionamento realizado por evento de captura gera conjuntos de teste otimistas que superestimam o desempenho dos modelos, o que pode levar à tomada de decisões equivocadas.

Em razão disso, foi especificado um conjunto de recomendações para construção de conjuntos de teste de acordo com o protocolo utilizado pelo projeto de armadilhas fotográficas. Caso os projetos considerem expansões, com a adição de novos pontos de captura, é recomendado que se utilize uma abordagem de particionamento por ponto de captura, uma vez que se verificou uma redução significativa na acurácia dos modelos quando utilizados em pontos de captura não incluídos no treinamento. Por outro lado, a abordagem de particionamento por tempo é aconselhada para os casos em que os modelos serão utilizados para fazer prendições apenas na mesma rede de câmeras utilizadas durante o treinamento. Nesse caso, o particionamento por tempo seria capaz de representar situações como mudança natural do plano de fundo ao longo

do tempo e espécies em pontos de captura nos quais não foram observadas durante o treinamento. Também foi verificado que o desempenho dos modelos decai ao longo do tempo, sendo recomendado que os projetos considerem um retreinamento periódico dos modelos de reconhecimento.

Foi verficado também que modelos pré-treinados na base ImageNet e com *fine tuning* em bases de armadilhas fotográficas de maior porte, como a Snapshot Serengeti, oferecem melhores resultados quando utilizadas como extratores de características para reconhecimento de espécies em bases de menor porte. No entanto, os experimentos não se mostraram conclusivos sobre a efetividade dessa transferência de aprendizado encadeada para os casos em que as bases de destino da transferência têm imagens suficientes para a realização de *fine tuning* em mais camadas. Quanto à transferência realizada a partir de modelos treinados do zero na Snapshot Serengeti, o desempenho foi significativamente inferior em todas as situações apesar de o treinamento original obter resultados compatíveis com a literatura. Assim, ao se fazer transferência de aprendizado de modelos treinados em bases de armadilhas fotográficas é interessante comparar com modelos pré-treinados em bases genéricas como a ImageNet.

6.2 Limitações

As limitações desta pesquisa estão relacionadas às bases de imagens utilizadas e ao poder computacional necessário para treinar arquiteturas mais complexas de redes profundas, como segue:

- Foi utilizada apenas a base Snapshot Serengeti para o estudo das abordagens de particionamento dos dados. Foi considerada a utilização de outras bases de armadilhas fotográficas de grande porte, como as disponibilizadas no trabalho de Willi et al. [2018], mas, que por não possuirem identificação dos pontos de captura das imagens e apresentarem alguns problemas referentes às datas de obtenção, optou-se por não utilizá-las.

- As bases de armadilhas fotográficas utilizadas contêm apenas o rótulo relativo ao problema de classificação, não contendo indicação de onde está o animal da cena. Isso limitou a utilização da técnica de aumento artificial de dados, pois a aplicação de operações mais severas poderia cortar o animal da imagem e fornecer ao modelo uma instância que contenha somente o plano de fundo, o que poderia prejudicar ainda mais o aprendizado.

- Não foram utilizadas arquiteturas mais potentes, como a ResNet152 e a InceptionResNetV2, devido o poder computacional necessário para treinar esses modelos do zero, seguindo protocolo experimental adotado.

6.3 Trabalhos futuros

Durante a coleta de imagens de armadilhas fotográficas, é obtida uma grande quantidade de imagens que não contêm animal na cena, como no Snapshot Serengeti, onde o índice de imagens sem animal é de aproximadamente 75%. Nesse caso, seria interessante o estudo de técnicas de subtração de plano de fundo em imagens de armadilhas fotográficas que permitissem identificar regiões candidatas a conter animal na cena. Com isso, a aplicação dos modelos de classificação em regiões mais restritas da imagem poderia permitir a observação de detalhes que são perdidos devido o redimensionamento da imagem para o tamanho de entrada da rede. Essa identificação de regiões candidatas também possilitaria estimar a posição dos animais nas cenas, o que permitiria uma maior flexibilidade para aplicação de aumento artificial de dados mais severo.

Outro problema que necessita de maior investigação é a predição de classes minoritárias. Dependendo do projeto de armadilhas fotográficas, as espécies de interesse podem ter pouquíssimas instâncias para o treinamento dos modelos. Uma possibilidade é a utilização de classificação hierárquica baseada na similaridade das espécies. Essa abordagem permitiria a combinação de bases de dados com espécies distintas. Assim, ainda que determinada espécie não exista no âmbito de um projeto, as imagens daquela classe poderiam ajudar o modelo a aprender o conceito do grupo taxonômico num nível mais alto na hierarquia. Outra vantagem seria que caso o modelo não tenha certeza quanto à predição da classe granular, poderia oferecer uma resposta em um nível superior da hierarquia. Dessa forma, determinadas imagens que o classificador não tivesse confiança na classificação poderiam ser marcadas para revisão manual.

Referências Bibliográficas

Ahumada, J. A.; Hurtado, J. & Lizcano, D. (2013). Monitoring the status and trends of tropical forest terrestrial vertebrate communities from camera trap data: a tool for conservation. *PloS one*, 8(9):e73707.

Akçay, S.; Kundegorski, M. E.; Devereux, M. & Breckon, T. P. (2016). Transfer learning using convolutional neural networks for object classification within x-ray baggage security imagery. Em *Image Processing (ICIP), 2016 IEEE International Conference on*, pp. 1057--1061. IEEE.

Burton, A. C.; Neilson, E.; Moreira, D.; Ladle, A.; Steenweg, R.; Fisher, J. T.; Bayne, E. & Boutin, S. (2015). Wildlife camera trapping: a review and recommendations for linking surveys to ecological processes. *Journal of Applied Ecology*, 52(3):675--685.

Changzhen, X.; Cong, W.; Weixin, M. & Yanmei, S. (2016). A traffic sign detection algorithm based on deep convolutional neural network. Em *Signal and Image Processing (ICSIP), IEEE International Conference on*, pp. 676--679. IEEE.

Chen, G.; Han, T. X.; He, Z.; Kays, R. & Forrester, T. (2014). Deep convolutional neural network based species recognition for wild animal monitoring. Em *Image Processing (ICIP), 2014 IEEE International Conference on*, pp. 858--862. IEEE.

Christodoulidis, S.; Anthimopoulos, M.; Ebner, L.; Christe, A. & Mougiakakou, S. (2017). Multisource transfer learning with convolutional neural networks for lung pattern analysis. *IEEE journal of biomedical and health informatics*, 21(1):76--84.

Donahue, J.; Jia, Y.; Vinyals, O.; Hoffman, J.; Zhang, N.; Tzeng, E. & Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. Em *International conference on machine learning*, pp. 647--655.

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

Elmahdy, M. S.; Abdeldayem, S. S. & Yassine, I. A. (2017). Low quality dermal image classification using transfer learning. Em *Biomedical & Health Informatics (BHI), 2017 IEEE EMBS International Conference on*, pp. 373--376. IEEE.

Gajapersad, K. (2018). Central suriname nature reserve data set. Data Set Identifier: TEAM-DataPackage-20181004123052_1315.

Goodfellow, I.; Bengio, Y. & Courville, A. (2016). *Deep learning*. MIT press.

He, K.; Zhang, X.; Ren, S. & Sun, J. (2016a). Deep residual learning for image recognition. Em *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770--778.

He, Z.; Kays, R.; Zhang, Z.; Ning, G.; Huang, C.; Han, T. X.; Millspaugh, J.; Forrester, T. & McShea, W. (2016b). Visual informatics tools for supporting large-scale collaborative wildlife monitoring with citizen scientists. *IEEE Circuits and Systems Magazine*, 16(1):73--86.

Heravi, E. J.; Aghdam, H. H. & Puig, D. (2017). Classification of foods by transferring knowledge from imagenet dataset. Em *Ninth International Conference on Machine Vision*, pp. 1034128--1034128. International Society for Optics and Photonics.

Hinton, G.; Srivastava, N. & Swersky, K. (2012). Rmsprop: Divide the gradient by a running average of its recent magnitude. *Neural networks for machine learning, Coursera lecture 6e*.

Horn, G. V.; Aodha, O. M.; Song, Y.; Cui, Y.; Sun, C.; Shepard, A.; Adam, H.; Perona, P. & Belongie, S. (2018). The inaturalist species classification and detection dataset. Em *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8769–8778.

Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. Em *International conference on machine learning*, pp. 448--456.

Joly, A.; Goëau, H.; Bonnet, P.; Spampinato, C.; Glotin, H.; Rauber, A.; Vellinga, W.-P.; Fisher, R. & Müller, H. (2014). Are species identification tools biodiversity-friendly? Em *Proceedings of the 3rd ACM International Workshop on Multimedia Analysis for Ecological Data*, pp. 31--36. ACM.

Kays, R.; Crofoot, M. C.; Jetz, W. & Wikelski, M. (2015). Terrestrial animal tracking as an eye on life and planet. *Science*, 348(6240):aaa2478.

Kays, R.; Kranstauber, B.; Jansen, P.; Carbone, C.; Rowcliffe, M.; Fountain, T. & Tilak, S. (2009). Camera traps as sensor networks for monitoring animal communities. Em *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*, pp. 811--818. IEEE.

Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A.; Sutskever, I. & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Em *Advances in neural information processing systems*, pp. 1097--1105.

LeCun, Y.; Bengio, Y. & Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436--444.

Lima, M. & Santos, F. (2018). Caxiuanã data set. Data Set Identifier: TEAM-DataPackage-20181004123052_1315.

Lin, M.; Chen, Q. & Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P. & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. Em *European conference on computer vision*, pp. 740--755. Springer.

Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT Press.

Norouzzadeh, M. S.; Nguyen, A.; Kosmala, M.; Swanson, A.; Palmer, M. S.; Packer, C. & Clune, J. (2018). Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences*, p. 201719367.

Oquab, M.; Bottou, L.; Laptev, I. & 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*, pp. 1717--1724.

Pan, S. J. & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345--1359.

Robbins, H. & Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pp. 400--407.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M. et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211--252.

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

Sokolova, M.; Japkowicz, N. & Szpakowicz, S. (2006). Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation. Em *Australasian joint conference on artificial intelligence*, pp. 1015--1021. Springer.

Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I. & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929--1958.

Swanson, A.; Kosmala, M.; Lintott, C.; Simpson, R.; Smith, A. & Packer, C. (2015a). Data from: Snapshot serengeti, high-frequency annotated camera trap images of 40 mammalian species in an african savanna. Dryad Digital Repository. https://doi.org/10.5061/dryad.5pt92.

Swanson, A.; Kosmala, M.; Lintott, C.; Simpson, R.; Smith, A. & Packer, C. (2015b). Snapshot serengeti, high-frequency annotated camera trap images of 40 mammalian species in an african savanna. *Scientific data*, 2.

Szegedy, C.; Ioffe, S.; Vanhoucke, V. & Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. Em *AAAI*, pp. 4278--4284.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V. & Rabinovich, A. (2015). Going deeper with convolutions. Em *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1--9.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J. & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818--2826.

Tabak, M. A.; Norouzzadeh, M. S.; Wolfson, D. W.; Sweeney, S. J.; VerCauteren, K. C.; Snow, N. P.; Halseth, J. M.; Di Salvo, P. A.; Lewis, J. S.; White, M. D. et al. (2018). Machine learning to classify animal species in camera trap images: applications in ecology. *Methods in Ecology and Evolution*.

Tajbakhsh, N.; Shin, J. Y.; Gurudu, S. R.; Hurst, R. T.; Kendall, C. B.; Gotway, M. B. & Liang, J. (2016). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging*, 35(5):1299--1312.

TEAM Network (2011). Terrestrial vertebrate protocol implementation manual, v. 3.1. Tropical Ecology, Assessment and Monitoring Network, Center for Applied Biodiversity Science, Conservation International, Arlington, VA, USA.

Villa, A. G.; Diez, G.; Salazar, A. & Diaz, A. (2016). Animal identification in low quality camera-trap images using very deep convolutional neural networks and confidence thresholds. Em *International Symposium on Visual Computing*, pp. 747--756. Springer.

Villa, A. G.; Salazar, A. & Vargas, F. (2017). Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks. *Ecological Informatics*, 41:24--32.

Willi, M.; Pitman, R. T.; Cardoso, A. W.; Locke, C.; Swanson, A.; Boyer, A.; Veldthuis, M. & Fortson, L. (2018). Identifying animal species in camera trap images using deep learning and citizen science. *Methods in Ecology and Evolution*.

Yosinski, J.; Clune, J.; Bengio, Y. & Lipson, H. (2014). How transferable are features in deep neural networks? Em *Advances in neural information processing systems*, pp. 3320--3328.

Yu, X.; Wang, J.; Kays, R.; Jansen, P. A.; Wang, T. & Huang, T. (2013). Automated identification of animal species in camera trap images. *EURASIP Journal on Image and Video Processing*, 2013(1):52.

Zhang, C.; Bengio, S.; Hardt, M.; Recht, B. & Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.