Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Aprendizado Ativo Efetivo e Eficiente para Análise de Imagens em Patologia Utilizando Aprendizado Profundo

André Lauar Sampaio Meirelles

Tese apresentada como requisito parcial para
conclusão do Doutorado em Informática

Orientador
Prof. Dr. George Luiz Medeiros Teodoro

Brasília
2022

Ficha Catalográfica de Teses e Dissertações

Está página existe apenas para indicar onde a ficha catalográfica gerada para dissertações de mestrado e teses de doutorado defendidas na UnB. A Biblioteca Central é responsável pela ficha, mais informações nos sítios:

http://www.bce.unb.br
http://www.bce.unb.br/elaboracao-de-fichas-catalograficas-de-teses-e-dissertacoes

Esta página não deve ser inclusa na versão final do texto.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Aprendizado Ativo Efetivo e Eficiente para Análise de Imagens em Patologia Utilizando Aprendizado Profundo

André Lauar Sampaio Meirelles

Tese apresentada como requisito parcial para
conclusão do Doutorado em Informática

Prof. Dr. George Luiz Medeiros Teodoro (Orientador)
PPGI/UnB

Prof. Dr. Bruno Luiggi Macchiavello Espinoza
PPGI/UnB

Prof. Dr. Renato Antônio Celso Ferreira
DCC/UFMG

Prof. Dr. Adriano Alonso Veloso
DCC/UFMG

Prof. Dr. Ricardo Pezzuol Jacobi
CIC/UnB

Prof. Dr. Ricardo Pezzuol Jacobi
Coordenador do Programa de Pós-graduação em Informática

Brasília, 17 de agosto de 2022

Dedicatória

A família nos mantém em foco e nos incentiva. Sem foco e dedicação não realizamos. Assim, dedico esse trabalho à Marianna e João, meus pais Luiz e Marisa e meus irmãos Humberto e Fernanda.

Agradecimentos

Agradeço a Deus por manter a mim e a minha família livres das doenças do mundo.

À equipe do Departamento de Ciência da Computação da UnB pela presteza de sempre, aos docentes que me acompanharam, especialmente à Profa. Célia Ghedini com suas contribuições durante as disciplinas de Seminários.

Ao meu orientador, Prof. Dr. George Luiz Medeiros Teodoro, que apostou em mim antes mesmo que eu tivesse a certeza das possibilidades do trabalho que iríamos realizar, enquanto ainda cursava disciplinas como aluno especial. Obrigado por me acompanhar durante todos esses anos, com muita paciência, em um tom de incentivo, franqueza e tranquilidade. Suas contribuições vão além do ganho acadêmico e cultural.

Agradeço ainda aos nossos colegas e colaboradores externos Prof. Dr. Joel Saltz e Dr. Tahsin Kurc, que disponibilizaram o acesso aos recursos computacionais para realização dos experimentos, sem os quais as dificuldades seriam enormes. Suas contribuições, não só com recursos mas com ideias e discussões contribuíram definitivamente para o avanço dos trabalhos. Destaco as contribuições de Rajarsi Gupta e agradeço por disponibilizar seu tempo testando e realizando anotações reais de imagens de histopatologia utilizando o DADA e a interface gráfica desenvolvida como resultado desse trabalho.

Agradeço aos colegas de pesquisa, Willian Barreiros pelas contribuições de ideias e auxílio com códigos, especialmente em um começo difícil, e Jeremias Gomes por dúvidas dirimidas, acadêmicas e pessoais.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

Modelos de aprendizado profundo demonstraram notável desempenho em tarefas de segmentação e classificação de imagens de patologia. Entretanto, esses modelos demandam grandes quantidades de dados anotados para seu treinamento. A geração dessa massa de dados em patologia é um processo intensivo em mão de obra, comprometendo muitas horas de trabalho por parte de patologistas experientes. O Aprendizado Ativo, ou *Active Learning (AL)*, oferece uma abordagem iterativa para a geração dessas bases de dados, reduzindo o custo das anotações. Nesse trabalho, foi proposta uma nova solução de aprendizado ativo, denominada *Diversity-Aware Data Acquisition (DADA)*, e foi avaliada sua efetividade na classificação baseada em *patches* de regiões de tecido de histopatologia. O DADA usa uma lógica de agrupamento que leva em consideração as características das imagens, extraídas de modelos de aprendizado profundo, e a incerteza preditiva desses modelos para selecionar exemplos de treinamento significativos. Além de produzir conjuntos de treinamento reduzidos, os custos de anotação também são diminuídos com ganhos de tempo de processamento, com o uso de uma solução de simplificação de CNNs também desenvolvida neste trabalho, o *Network Auto-Reduction (NAR)*. Com o NAR, tanto o custo de cálculo das incertezas preditivas, quanto de treinamento de modelos, são fortemente reduzidos. Adicionalmente, para viabilizar a utilização da solução na prática, uma interface gráfica Web foi adaptada para uso com o DADA. O DADA e o NAR foram avaliados experimentalmente sobre uma coleção de imagens de tecido cancerígeno e demonstraram que: (i) são selecionados *patches* que aceleram o processo de treinamento ao reduzir o número deles necessários para se atingir um dado nível de *Area Under the Curve (AUC)*; (ii) com o uso de *subpooling* o DADA apresenta significativa redução dos tempos de cada iteração de aquisição; e (iii) a combinação do DADA com NAR traz os tempos de execução de cada iteração a patamares práticos, mantendo a capacidade preditiva dos modelos de *deep learning* alvo. A generalização tanto do DADA quanto do NAR a outros contextos e aplicações são trabalhos futuros previstos, incluindo áreas como sensoriamento remoto e problemas de segmentação.

Palavras-chave: Aprendizado ativo, patologia, CNNs

Abstract

Deep learning methods have demonstrated remarkable performance in pathology image segmentation and classification tasks. However, these models require a large amount of annotated training data. Training data generation is a labor intensive process in digital pathology, often requiring substantial time commitment from expert pathologists. Active learning (AL) offers an iterative approach to generate training data needed by deep learning models, reducing the cost of manual data annotation. In this work, a new AL acquisition method, named *Diversity-Aware Data Aquisition* (DADA), is proposed and evaluated regarding its effectiveness in patch-based detection and classification of tissue image regions. The proposed method uses a clustering logic that takes into account image features, extracted from the deep learning model being trained, and model prediction uncertainty to select meaningful training samples (image patches). Besides reducing training set sizes, annotation costs are also diminished by computation time gains using a CNN simplification solution also developed in this work, the Network Auto-Reduction (NAR). With NAR, both uncertainty calculation costs and model training times are strongly reduced. Additionally, to make these solutions viable in practice, a Web based graphical interface was adapted to be used with DADA. The DADA/NAR solutions were experimentally evaluated with a collection of cancer tissue images and are able to: (i) select image patches that accelerate the training process by reducing the number of patches required to attain a given Area Under the Curve (AUC) value; (ii) using a subpooling approach, DADA dramatically reduces iteration times needed to select a new annotation set; and (iii) the combination of DADA and NAR brings down the execution times even more, reaching practical levels while keeping the predictive capacity of models. The generalisation of both DADA and NAR to other contexts and applications are expected future work, including application in areas such as remote sensing and image segmentation problems.

Keywords: Active Learning, pathology, CNNs

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AL Active Learning.

AUC Area Under the Curve.

BALD Bayesian Active Learning by Disagreement.

CNN Convolutional Neural Network.

DA Data Augmentation.

DADA Diversity-Aware Data Acquisition.

DL Deep Learning.

DQA DADA QuickAnnotator.

FLOP Floating Point Operations.

GPU Graphics Processing Unit.

IA Inteligência Artificial.

LIT Linfócito Infiltrante de Tumor.

NAR Network Auto-Reduction.

PCA Principal Component Analysis.

QA QuickAnnotator.

SVM Support Vector Machine.

TCGA The Cancer Genome Atlas.

UnB Universidade de Brasília.

VR Variation Ratios.

WSI Whole Slide Image.

Capítulo 1

Introdução

A IA, desde sua concepção, sempre foi tida como uma área de conhecimento com enorme capacidade de alterar as relações de trabalho e o modo de vida das pessoas, de maneiras ainda não totalmente compreendidas [1]. Com foco específico na aplicação da IA em processos de trabalho, uma corrente de pensamento [2] considera que a integração entre automação e IA com os processos atualmente empregados geraria os melhores frutos, produzindo expressivo aumento de produtividade mas permitindo ao operador humano fazer considerações críticas, baseadas em julgamentos onde a experiência e conhecimento são fundamentais, além de se manter uma cadeia de responsabilização.

Oportunidades de integração e sinergias existem em diferentes áreas do conhecimento, sendo uma realidade em diferentes contextos, com ferramentas comerciais já disponíveis em aplicações desde o atendimento a clientes até a medicina [3]. Entre as diversas especialidades médicas, a área da patologia clínica passa por grandes avanços tecnológicos, especialmente a partir do desenvolvimento e disseminação de *scanners* de alta resolução e da digitalização das imagens de tecido biológico. As imagens geradas nesse processo são denominadas *Whole Slide Images (WSIs)* e normalmente possuem tamanho da ordem de bilhões de pixels [4, 5]. A elevada resolução das imagens captura em detalhes a morfologia celular de tecidos, vascularização, presença de células inflamatórias e necroses, entre outras estruturas significativas, criando grandes oportunidades de ganhos para o tratamento, visto que a evolução da doença é, em grande parte, determinada a partir das modificações morfológicas observadas.

Apesar dos avanços trazidos pela digitalização das imagens, o exame manual de cada uma delas não possui a escalabilidade necessária para se extrair todas as informações disponíveis, além de demandar extenso trabalho por parte de profissionais qualificados, cuja mão de obra possui alto custo. Nesse contexto, a introdução de ferramentas automáticas de análise de imagens a partir de métodos de inteligência artificial traria benefícios qualitativos e quantitativos, ao permitir que o patologista analise múltiplas imagens em um

menor tempo, bem como a extração de maior quantidade de informações de cada uma.

Na atualidade, *deep learning* [6] é uma das subáreas da IA que tem atraído mais atenção, especialmente em problemas relacionados à análise e interpretação de imagens. Não por acaso, diversos estudos já foram desenvolvidos para aplicações médicas utilizando-se de soluções de *deep learning* [7, 8, 9, 10, 11, 12, 13]. Todavia, o desenvolvimento de soluções de aprendizado de uma forma geral e, em especial na área médica, esbarra nas dificuldades de se obter um conjunto de dados em quantidade e qualidade necessários para se treinar modelos de *deep learning* e obter resultados satisfatórios [14].

Existem variadas abordagens para se obter um conjunto de treinamento adequado. Uma delas é o esforço colaborativo (*crowdsourcing*) [15, 16, 17, 18]. Em aplicações em que não se demanda conhecimentos específicos para a produção de anotações nas imagens é possível contar com a colaboração do público em geral. Todavia, é necessário se utilizar de um controle de qualidade robusto, bem como infraestrutura para que um grande número de participantes possa atuar no trabalho.

Na área médica as anotações usualmente devem ser feitas por um especialista, limitando sobremaneira os possíveis benefícios desse tipo de abordagem e aumentando os custos. Em [15], por exemplo, um esforço colaborativo contou com a participação de 25 voluntários, entre médicos experientes e estudantes, gerando mais de 20.000 regiões anotadas para segmentação. Nesse trabalho, conforme os autores, foi necessário lidar com questões relativas à discordância entre as anotações de diferentes voluntários e processos de revisão. O foco foi em avaliar a qualidade das anotações realizadas conforme o processo proposto, bem como o nível de discordância entre anotações, não tendo sido mensurado o custo de tempo através de médias confiáveis para a realização das anotações por cada participante. Além disso, do ponto de vista qualitativo, as regiões anotadas foram pré-selecionadas, de maneira que os slides não foram analisados em sua completude.

Outra abordagem seria a geração de bases de treinamento sintéticas [19, 20, 21]. A geração de dados sintéticos reduz a dependência em anotações manuais, todavia, o desenvolvimento de métodos para produção de dados sintéticos realistas é ainda uma área de pesquisa muito ativa e em evolução. Uma terceira abordagem para o problema de geração de bases de dados é o uso de técnicas de aprendizado ativo (*Active Learning (AL)*).

O aprendizado ativo é um processo iterativo que busca aumentar a capacidade de aprendizado de um modelo a partir da seleção de pontos de treinamento com maior potencial informativo, de modo a se reduzir o total de exemplos de treinamento necessários para que o modelo atinja um determinado desempenho e, por consequência, reduzir os esforços de anotação demandados. As técnicas de AL buscam utilizar o conhecimento parcial do modelo, durante cada etapa de refinamento, de forma que este indique quais

pontos, de um conjunto de pontos disponíveis, poderiam trazer maiores ganhos de aprendizado [22].

A utilização de AL em problemas relacionados à patologia já foi foco de outros trabalhos [23, 24]. Entretanto, cada aplicação apresenta particularidades que fazem com que técnicas de AL utilizadas em uma situação não necessariamente produzam o mesmo nível de desempenho observado em outras. Como será demonstrado no Capítulo 7, técnicas desenvolvidas para outros domínios apresentaram resultados insatisfatórios quando aplicadas em patologia digital e, em especial, na aplicação motivadora utilizada nesse trabalho. Aspectos relativos ao tempo de processamento, bem como à forma de seleção dos pontos de treinamento, devem ser levados em consideração para que uma solução adequada seja atingida.

1.1 Definição do Problema

Aplicações voltadas para a área médica normalmente não possuem vastos bancos de dados de treinamento para que técnicas de *deep learning* possam ser aplicadas com sucesso. Para permitir uma adoção dessas técnicas de maneira mais ampla é necessário que sejam criados mecanismos que diminuam os custos de geração desse tipo de banco de dados, levando em consideração o número de imagens necessárias para se atingir um nível de acurácia[1] desejado. O aprendizado ativo é uma técnica viável e que já demonstrou resultados positivos em outras aplicações, todavia sua utilização direta dentro do contexto da patologia apresenta problemas, requerendo análises e adaptações para atingir os níveis de acurácia necessários para aplicações da medicina, bem como seus custos computacionais devem ser minimizados.

Assim, considera-se que uma solução de AL desenvolvida com foco nas aplicações de patologia deve ser capaz de produzir bases de dados de treinamento de tamanho menor que soluções genéricas e, ainda assim, sendo capaz de permitir o treinamento de modelos de *deep learning* com capacidade de classificação superior a essas soluções genéricas.

1.2 Objetivos

Desenvolver e avaliar um modelo de AL com foco em aplicações para patologia clínica, que leve em consideração as particularidades geradas pela análise de WSIs, minimizando tanto os custos computacionais da seleção quanto o custo de anotação dos exemplos de treinamento, a partir da obtenção de um menor número de pontos de treinamento possível. A solução deve ser inicialmente voltada a uma aplicação específica, descrita no

[1]Nesse contexto, acurácia se refere a um nível de desempenho de um modelo de classificação.

Capítulo 5, podendo ser generalizada a outras aplicações no futuro.

Objetivos específicos:

1. Desenvolver e analisar uma metodologia de aprendizado ativo a partir de modelos de *deep learning*, especificamente através de *CNNs*, que possa ser adequada a aplicações de patologia clínica, inicialmente com foco na aplicação motivadora definida no Capítulo 5;
2. Projetar e implementar um sistema integrado que permita a execução de diferentes estratégias de AL, bem como o uso de diferentes arquiteturas de redes neurais convolucionais;
3. Propor técnicas e novas otimizações para minimizar o tempo necessário para cada iteração de aquisição de imagens de treinamento, impedindo atrasos para que o especialista receba essas imagens para anotação;
4. Avaliar o resultado obtido em dados reais de tecidos cancerosos, com relação ao número de imagens necessárias para atingir determinado desempenho de classificação e o tempo necessário em todo o processo;
5. Desenvolver uma plataforma para simplificar o uso das técnicas, otimizações e estratégias propostas nesse trabalho.

1.3 Contribuições

Como resultado deste trabalho, três diferentes soluções foram desenvolvidas que, de maneira conjunta, permitem o uso prático da ferramenta de aprendizado ativo que será apresentada no decorrer deste documento.

- A primeira solução, *DADA* é a proposta de aprendizado ativo voltada à patologia, especificamente desenvolvida para identificar a presença de determinado tipo de célula em meio ao tecido de biópsias.
- A segunda solução é uma técnica de simplificação de *CNNs* que ao ser utilizada junto ao DADA reduz drasticamente o tempo necessário para que se selecione um conjunto de imagens para receber anotações.
- A terceira solução é uma interface gráfica Web que aplica a combinação DADA/NAR e expõe a um especialista as imagens que devem receber rótulos de uma maneira simples e rápida. Apesar do presente trabalho não ter foco em desenvolvimento

de interfaces como um produto de pesquisa, a disponibilização dessa ferramenta permite a continuidade da pesquisa ora conduzida, com a expansão do DADA a outras aplicações de patologia.

Essas contribuições produziram publicações em periódicos conforme abaixo, bem como um terceiro artigo, em fase final de escrita:

- DADA: "*Effective Active Learning in Digital Pathology: A Case Study in Tumor Infiltrating Lymphocytes*" [25]
- NAR: "*Building Efficient CNN Architectures for Histopathology Images Analysis: A case-study in Tumor-Infiltrating Lymphocytes*" [26]

1.4 Organização da Tese

O restante desse trabalho segue a seguinte organização:

1. **Capítulo 2** [**Aprendizado de Máquina**]: aborda conceitos e teoria acerca da disciplina de aprendizado de máquina, detalhando algumas das técnicas mais utilizadas na literatura, bem como as características específicas das técnicas de *deep learning* e as razões de sua adequação ao problema em estudo;
2. **Capítulo 3** [**Aprendizado Ativo**]: apresenta o conceito de aprendizado ativo e as técnicas mais utilizadas para sua aplicação utilizando modelos de aprendizado clássico e modelos de *deep learning*;
3. **Capítulo 4** [**Trabalhos Relacionados**]: detalha trabalhos relevantes, relacionados ao tema em estudo e aplicados a AL e a redes convolucionais, bem como as técnicas de simplificação de CNNs mais destacadas;
4. **Capítulo 5** [**Aplicação Motivadora**]: descreve a aplicação foco do trabalho desenvolvido, detalhando as particularidades inerentes à análise de WSIs e como um banco de dados foi formado para permitir a execução dos experimentos ora descritos;
5. **Capítulo 6** [**Diversity-Aware Data Acquisition (DADA)**]: apresenta a proposta de aprendizado ativo voltada para a análise de WSIs, dentro da aplicação motivadora já descrita;
6. **Capítulo 7** [**Resultados**]: descreve os resultados experimentais obtidos, analisando como esses resultados se posicionam frente a abordagens relacionadas e apresentadas anteriormente;
7. **Capítulo 8** [**Conclusão**]: discute os resultados obtidos, a consecução dos objetivos e as questões ainda abertas.

Capítulo 2

Aprendizado de Máquina

Aprendizado de máquina pode ser definido a partir de duas visões complementares. Uma vertente mais genérica [27] considera o aprendizado de máquina como sendo a capacidade de um sistema aprender a partir de suas próprias experiências, compreendendo seus erros, ajustando seu comportamento e aprimorando a maneira de executar determinada tarefa, de modo que os resultados produzidos sejam melhorados com o tempo.

Em uma visão mais sistemática, o aprendizado de máquina é a capacidade de prever um resultado a partir de características (*features*) de um exemplo ou conjunto de exemplos fornecidos como entrada para um sistema [28]. O processo preditivo se dá a partir da observação e correlação entre as características de um conjunto de treinamento e o resultado correspondente para cada exemplo, nesse caso denominado aprendizado supervisionado. A partir desse conjunto de entradas/saídas é possível construir modelos que reconheçam a correlação entre as características e resultados e produzam uma previsão, dada uma entrada até então desconhecida.

Outras formas de aprendizado existem, com abordagens semi-supervisionadas e não supervisionadas. Nesses casos, os dados estão parcialmente rotulados (semi-supervisão) ou pode não haver rótulo algum, de modo que se desconheça completamente qual seria a saída atribuída a cada entrada (não supervisionado). O trabalho aqui desenvolvido foca nos modelos de aprendizado supervisionado.

Usualmente, o relacionamento entre as características sobre a qual a previsão é feita (X) e a previsão propriamente dita pode ser representado como uma função $f(X)$. De maneira genérica, o aprendizado de máquina supervisionado pode ser formalizado como a procura por uma função $\hat{f}(X)$ que aproxima a função $f(X)$, descrevendo um modelo estatístico $\hat{f}(X) \approx f(X) + \epsilon$, que considera a presença de um erro aleatório ϵ tal que a expectativa desse erro tenda a zero ($E(\epsilon) \to 0$) [28]. Diferentes modelos de previsão já foram desenvolvidos para se definir a função que melhor aproxima o modelo estatístico em estudo, como: árvores de decisão [29], florestas de decisão [30], regressores logísticos [31,

32], *Support Vector Machines (SVMs)* [28, 33], redes neurais (RNs) [34, 35], classificadores Bayesianos [36] entre outros [37].

Deep learning se enquadra como uma especialização das redes neurais clássicas (*feed-forward neural networks*) [6]. As ferramentas de DL se baseiam em múltiplas representações dos dados de entrada, organizadas de maneira hierárquica em camadas. Um dos diferenciais dessas técnicas é que as características que representam as entradas são aprendidas durante o treinamento, bem como a correlação entre elas, utilizada para a classificação, ao invés de serem elaboradas e extraídas separadamente em uma fase anterior à fase de previsão.

Basicamente, os preditores disponíveis na literatura e que não se enquadram no conceito de modelos de *deep learning* são considerados como técnicas clássicas ou convencionais (Figura 2.1). Nesses casos, é necessário um esforço de definição de quais *features* melhor representam os dados, o que usualmente demanda conhecimentos específicos acerca da natureza dos mesmos.

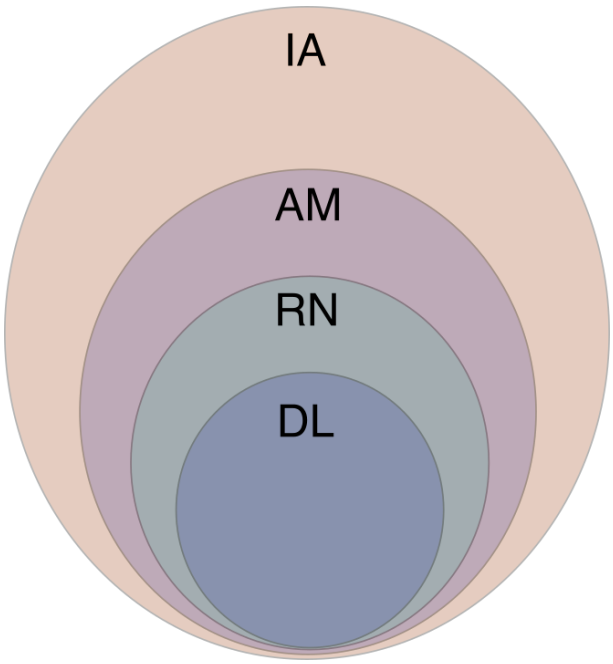


Figura 2.1: Organização entre as disciplinas de IA. Aprendizado de Máquina (AM), Redes Neurais (RN) e *DL*. Adaptado de [38].

Diversos problemas podem ser resolvidos através da aplicação de modelos preditivos e do aprendizado de máquina, sendo os mais comuns problemas de classificação e de regressão. Áreas específicas, como a visão computacional, aplicam modelos preditivos a outros problemas, como a detecção de objetos e a segmentação. O problema de classificação procura atribuir um valor qualitativo para cada conjunto de valores de entrada apresentado a um preditor, efetivamente dividindo os itens que compõem o universo de estudo em

um número finito de classes. A regressão, por sua vez, produz como resultado um valor quantitativo, usualmente dentro de um intervalo contínuo.

Nas seções seguintes será apresentado um breve resumo das técnicas de aprendizado de máquina clássicas mais importantes da literatura, bem como exemplos de suas aplicações na área médica, de modo que as principais diferenças entre essas e os modelos de *deep learning* sejam evidenciados. O problema focal da seleção apresentada é o de classificação, que é aquele abordado nesse trabalho.

2.1 Técnicas Clássicas de Aprendizado de Máquina

As técnicas de aprendizado convencionais dependem de uma fase prévia de extração de características dos pontos ou dados de entrada para que o processo de treinamento possa ocorrer. Isso se deve ao fato de que as ferramentas de aprendizado não possuem mecanismos de representação de informação próprios e não têm capacidade de operar sobre os dados em sua forma original. A extração de *features* é um processo que pode demandar extenso trabalho por parte de um especialista, podendo apresentar significativo impacto no desempenho de classificação caso as características selecionadas não representem bem o conjunto de dados objeto do estudo [39, 40, 41, 42, 43, 44, 45, 46].

As Seções 2.1.1 a 2.1.4 apresentam exemplos de abordagens clássicas aplicadas a problemas de medicina por imagem, preferencialmente casos relativos à patologia clínica. As abordagens destacadas ilustram os pontos de principal relevância para o trabalho aqui desenvolvido, não correspondendo a uma conjunto exaustivo de publicações.

2.1.1 Regressão Logística

Regressões logísticas são utilizadas em problemas de classificação, principalmente nas situações em que existem apenas duas classes possíveis (classificação binária). Essencialmente, a regressão logística busca modelar múltiplas relações lineares de uma variável dependente (a previsão) dado um conjunto de variáveis independentes (X_i). A modelagem é definida pela função logística $f(z)$, conforme a equação 2.1.

$$f(z) = \frac{1}{1 + e^{-z}};\ z = \alpha + \sum \beta_i X_i \tag{2.1}$$

Uma das principais características da regressão logística é que a função $f(z)$ produz resultados no intervalo $[0, 1]$, podendo ser interpretados como uma probabilidade. Considerando o caso da classificação binária, a classe de um exemplo de entrada, definido por um vetor de variáveis independentes $X = [X_1, X_2, ..., X_k]$, pode ser interpretada como:

$$
\begin{aligned}
&P(C=1|X) = \frac{1}{1+e^{-(\alpha+\sum \beta_i X_i)}},\ dado\ \ 0 \leq i \leq k; \\
&P(C=0|X) = 1 - P(C=1|X).
\end{aligned}
\tag{2.2}
$$

O treinamento de um modelo de regressão logística envolve a estimativa de valores para os parâmetros α e β_i, resultando em um total de parâmetros igual a $k+1$, em que k corresponde ao número de variáveis independentes.

A regressão logística é frequentemente utilizada em trabalhos voltados à medicina, como aqueles desenvolvidos em [47, 48]. Os trabalhos citados apresentam uma comparação entre os resultados obtidos por um modelo de regressão logística com aqueles apresentados por um modelo de rede neural clássica. A partir do exposto nesses trabalhos, é interessante notar a dificuldade na seleção das *features* que seriam utilizadas para o treinamento e previsão dos modelos, usualmente demandando uma análise de parâmetros em diferentes configurações desses modelos para se estabelecer um conjunto otimizado.

Apesar do resultado obtido por ambos os modelos desenvolvidos por *Borque et al.* [47] ser semelhante, a complexidade computacional favorece o uso do modelo logístico considerado. Todavia, o custo de construir um conjunto de características ideal e a adição de mais características favoreceria o modelo de rede neural, como concluem os próprios autores. Em um estudo semelhante ao de *Borque et al.*, mas apresentando um maior número de casos e um conjunto diferente de características, *Han et al.* [48] produziram um modelo de rede neural clássico que obteve resultados melhores que o modelo de regressão logística popularmente utilizado à época.

Os trabalhos discutidos mostram como a seleção manual das características utilizadas para representar o conjunto de dados disponível afeta o resultado da classificação. Adicionalmente, a dificuldade em se selecionar essas características é substancial, apesar de muitas vezes não ser destacada nos trabalhos.

2.1.2 Árvore de Decisão e Florestas de Decisão

As árvores de decisão são usualmente consideradas ferramentas de mineração de dados [49], que podem ser convertidas em modelos de classificação ou regressão [50, 51, 52]. Esse tipo de modelo procura dividir o espaço amostral em diferentes regiões, considerando um conjunto de características de entrada $X_i = [x_1, x_2, ..., x_m]$ contendo m variáveis para cada exemplo X_i, criando pontos de decisão em relação às variáveis. Ao fazer isso, a classificação de um item de entrada pode ser dada a partir do caminho de decisões tomadas desde a raiz da árvore até uma folha, que indicará a qual classe esse item pertence [28].

A criação da árvore depende da identificação das variáveis x_k envolvidas em cada ponto de decisão k e do valor de referência para cada ponto s_k (Figura 2.2). Quanto

mais profunda a árvore menores são as regiões amostrais, o que pode favorecer o fenômeno de *overfitting*. Todavia, árvores muito rasas podem não captar relações intrínsecas entre os dados. Usualmente as árvores devem passar por um processo de poda depois de construídas para eliminar nós pouco significativos. Algoritmos como o *Classification and Regression Trees (CART)* [53], C4.5 [54] e C5.0 [55] são popularmente utilizados tanto para a construção das árvores como para o processo de poda, quando necessário.

Uma abordagem diferente é apresentada em [52]. Nesse trabalho, as árvores de decisão são geradas a partir de um conjunto de regras de associação. Nesse caso, além dos esforços de definição e extração de *features*, é necessária uma fase de otimização do classificador, com a escolha de um conjunto de regras que produzam os melhores resultados para o conjunto de dados em mãos.

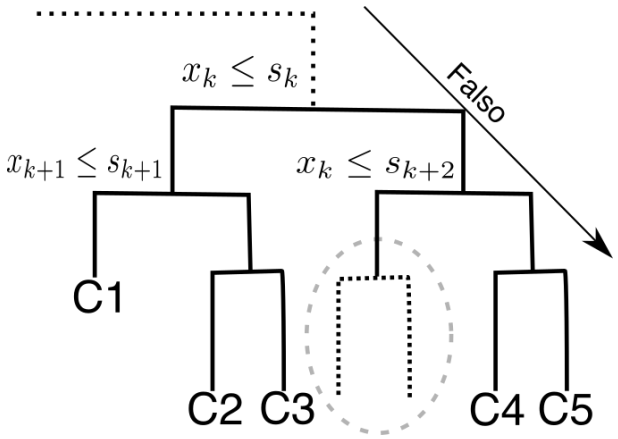


Figura 2.2: Árvore de decisão. O ramo destacado pela elipse com a linha hachurada é gerado a partir de um ponto de decisão possível.

As florestas de decisão surgem posteriormente como uma melhoria que combina diferentes árvores de decisão para produzir um resultado final, sendo, portanto, uma forma de *ensemble* (ver Seção 6.1). Existem diferentes abordagens para a criação de uma floresta de decisão, como AdaBoost [56] e *Random Forest* [57], bem como maneiras de combinar os resultados individuais de cada árvore, com o uso de pesos ou o uso de meta-aprendizado [30].

Um dos trabalhos que usa árvores de decisão em aplicações de medicina por imagem é apresentado em [51], por *Kuo et al.*. Assim como ocorre com as soluções baseadas em regressão logística, esse trabalho demanda esforço considerável para a extração de *features*. No trabalho mencionado, especial dificuldade é encontrada na extração de *features* de textura, visto que a aplicação foco utilizava imagens de ultrassom, que naturalmente são compostas por escalas de cinza. Os resultados apresentados foram consistentes, superando a de um médico experiente que analisou os mesmos exames de testes. Todavia, a metodologia desenvolvida ainda dependia de uma etapa de anotação manual prévia sobre as imagens, com a seleção de uma região de interesse.

2.1.3 *Support Vector Machines*

Os classificadores do tipo SVM são aplicados a problemas em que os itens se distribuem em duas classes, de maneira que as características analisadas podem ser separadas por limites lineares. Os SVMs buscam identificar os possíveis hiperplanos de separação dos exemplos e define como plano de separação aquele que maximiza as margens entre o hiperplano e os pontos mais próximos dele (vetores de suporte). Dessa forma, procura definir um critério de classificação que produza a separação mais clara entre os pontos de treinamento disponíveis, o que traz como benefício uma melhor capacidade de generalização [28, 33].

Para classificar pontos que não possuem uma separação linear possível, o SVM utiliza duas propriedades, a margem branda (*soft margin*), que permite que um certo número de pontos possa permanecer do lado incorreto de classificação, dado pelo hiperplano de separação dos pontos, e as funções de *kernel*. Essas funções projetam dados de um espaço de menor dimensão em espaços de dimensões superiores, em que, possivelmente, possam ser linearmente classificados, utilizando do mesmo método a partir dos hiperplanos.

À princípio, os classificadores SVM se aplicam a problemas de classificação binária, mas podem ser generalizados para aplicações com múltiplas classes. Em [58], *Bhatia et al* apresentam uma solução para a classificação de 4 classes de doenças cardíacas ou a ausência de cardiopatias, utilizando um SVM. Como é comum com as demais soluções tradicionais de classificação, a identificação das *features* utilizadas para alimentar o classificador é um problema. Nesse caso, os autores utilizaram um algoritmo genético para buscar aquelas que otimizariam o desempenho de classificação.

Já em [59], *Corredor et al.* utilizam um classificador SVM para identificar a presença de núcleos celulares de linfócitos em tecido canceroso do pulmão. Adicionalmente, os autores fazem uma comparação entre os resultados obtidos entre o modelo SVM e um modelo de *deep learning*, tanto em relação ao F-Score obtido quanto ao tempo de processamento e número de exemplos de treinamento necessários. O principal objetivo do trabalho foi buscar um conjunto de características discriminantes para a identificação de linfócitos. Apesar dos resultados obtidos a partir das características selecionadas ter sido satisfatório, é difícil estabelecer uma comparação entre os dois métodos empregados pois as metodologias de avaliação são distintas. O tempo de processamento necessário para treinar um modelo SVM é, todavia, consideravelmente menor do que o demandado para modelos de DL.

2.1.4 Redes Neurais

As redes neurais clássicas (RNC) foram desenvolvidas a partir dos trabalhos publicados por *McCulloch e Pitts* [34], *Rumelhart et al.* [35] e *Widrow e Hoff* [60]. Esses modelos,

também conhecidos como *feedforward networks*, se baseiam em um encadeamento de funções lineares e não lineares, de forma que a saída de cada uma é fornecida como entrada para a função seguinte, formando camadas individuais com processamento sequencial. Obrigatoriamente as redes apresentam uma camada de entrada e podem apresentar uma camada de saída e camadas ocultas, posicionadas entre as camadas anteriores.

As redes neurais são baseadas no modelo de um nó de processamento conhecido como "neurônio". Esse modelo, inicialmente formulado por *McCulloch e Pitts*, é ilustrado na Figura 2.3. Cada neurônio recebe como entrada um conjunto de sinais ($S = \{s_1, ..., s_j\}$), que são inicialmente combinados em uma função linear ($h_k(S)$, conforme equação 2.3).

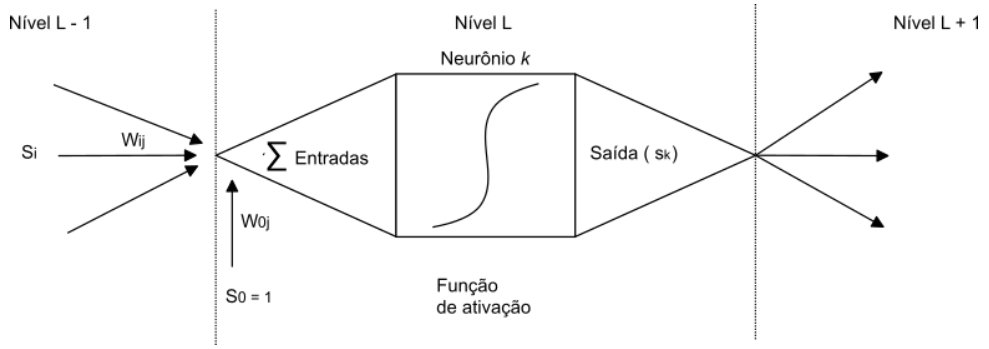


Figura 2.3: Modelo de neurônio utilizado nas redes neurais clássicas.

$$h_k(S) = w_{0,k} + w_{1,k}s_{1l-1} + ... + w_{n,k}s_{nl-1} = w_{0,k} + \sum_{i=1} w_{i,k}s_{il-1}, \tag{2.3}$$

onde w_{ik} corresponde aos pesos atribuídos a cada sinal de entrada i do neurônio k e $w_{0,k}$ corresponde a um viés (*bias*).

No modelo clássico, os sinais de entrada são valores binários, representando a "ativação" ou não de cada neurônio. A saída produzida por cada nó é então dada a partir da aplicação de uma função não linear sobre a função $h_k(S)$, denominada função de ativação, usualmente uma função onda quadrada, conforme exposto:

$$s_{kl} = \sigma_k = \mathcal{A}(h_k(S)) \ = \ \mathcal{A}\left(w_{0,k} + \sum_{i=0}^{n} w_{i,k}s_{il-1}\right), \tag{2.4}$$

onde σ_k corresponde ao sinal produzido pelo neurônio k, na camada l e $\mathcal{A}$ é a função de ativação utilizada. Nessa construção, cada neurônio recebe como entrada a saída de todos os neurônios da camada anterior, com pesos distintos para cada conexão.

Existem diversos exemplos de uso de redes neurais clássicas em aplicações médicas na literatura, como os trabalhos desenvolvidos em [61, 62]. *Arbach et al.* [61] utilizam redes neurais para classificar regiões suspeitas de imagens de ressonância magnética em exames de mama para classificar lesões como malignas ou benignas. Além das usuais dificuldades

em extrair características relevantes para serem passadas ao classificador, a abordagem apresentada demanda ainda uma anotação prévia e manual de regiões de interesse, que posteriormente passarão por um processo de segmentação.

Já *Das et al.* [62] apresentam uma solução baseada em um *ensemble* de redes neurais clássicas para classificar registros de pacientes contidos em um banco de dados. Os *ensembles* são definidos em maiores detalhes na Seção 3.5.1 mas, de maneira geral, correspondem a classificadores gerados pela combinação de modelos treinados sobre o mesmo domínio. No total, até 76 atributos relacionados a doenças coronárias são usadas para descrever cada entrada e para classificar os pacientes como potenciais portadores de doenças do coração. O *ensemble* criado é composto de 3 redes neurais com pesos iniciais selecionados aleatoriamente. Esse modelo obteve acurácia de 89,01% sobre a base de testes utilizada, superando classificadores disponíveis à época, como regressores logísticos, árvores de decisão e classificadores bayesianos.

Uma característica em comum a ambos os trabalhos e também presente em grande número de publicações na literatura [63, 64, 65] é o tamanho das redes utilizadas. Nos dois casos descritos as redes utilizadas eram compostas de apenas 3 camadas, incluindo as camadas de entrada e saída. Essa constatação reflete as limitações presentes à época para o treinamento desse tipo de rede, considerando o grande número de pesos que redes mais profundas produziriam, conforme será descrito em maiores detalhes na Seção 2.2.

2.2 Aprendizado Profundo (*Deep Learning*)

Desenvolvimentos teóricos já mostraram no passado que redes neurais clássicas com 2 camadas ocultas [66] ou mesmo uma única [67] seriam capazes de aproximar qualquer função não linear para problemas de classificação. A aproximação dada pelas redes neurais se aplica a qualquer função contínua, de um espaço de dimensão finita para outro, com acurácia arbitrária, dependente do número de nós presentes na camada oculta.

A partir das demonstrações teóricas apresentadas, o desenvolvimento de redes com maior profundidade se mostrava aparentemente inócuo. Todavia, a prática demonstra que a aplicação desses conceitos não é simples e a profundidade das redes possui impacto expressivo nos resultados [68]. Adicionalmente, apesar de focarem no número de camadas, as teorias desenvolvidas dependem de um número muito grande de neurônios nas camadas ocultas para criar as aproximações desejadas, o que produziria uma quantidade de pesos demasiadamente elevada, podendo tornar o processo de treinamento impraticável.

A partir do desenvolvimento do algoritmo de treinamento por gradientes descendentes e suas variantes, que utilizam a técnica de *backpropagation* [69, 70], passou a ser possível

treinar redes neurais contendo camadas ocultas. Assim, redes mais profundas (com maior número de camadas) eram teoricamente possíveis, o que remete ao termo *deep learning*.

Na atualidade, as redes consideradas soluções de *deep learning* além de apresentar múltiplas camadas, possuem a capacidade de formar representações parciais dos dados, através de características aprendidas durante o treinamento, eliminando a necessidade de fases de pré-processamento e extração de *features* manualmente selecionadas.

O desenvolvimento de algoritmos que possibilitam o treinamento de redes com múltiplas camadas, e que podem chegar a possuir dezenas ou centenas de milhões de pesos, permitiu que esses modelos adquirissem grande poder de representação. Todavia, o processo de aprendizagem demanda grandes volumes de dados e o treinamento de redes dessa magnitude, no início dos anos 1990, era inviável. Trabalhos iniciais, como aquele desenvolvido por *Le Cun et al.* [71] no reconhecimento de dígitos manuscritos, reportam que cerca de 3 dias eram necessários para que uma rede com 4 camadas fosse treinada sobre um conjunto de 7.291 imagens de 16 x 16 pixels.

Dada a impossibilidade prática de se treinar redes grandes o suficiente nesse período, as técnicas de *deep learning* passaram por um momento de dormência. A partir do desenvolvimento e utilização de unidades de processamento gráfico (*Graphical Processing Units - GPUs*) para o treinamento desse tipo de rede, o processo passou a ser realizado de maneira muito mais rápida, fazendo com que, a partir de 2006, as soluções de *deep learning* renascessem [72] no meio científico.

2.2.1 Redes Neurais Convolucionais

As redes neurais convolucionais (*Convolutional Neural Networks (CNNs)*) [71, 73] são um modelo de rede muito comum dentro dos modelos de *deep learning* usualmente utilizados na literatura. As CNNs, apesar de aceitarem múltiplos tipos de entradas, são voltadas pra trabalhar com imagens, de forma que ao se tratar desse tipo de rede, assume-se que a entrada seja uma matriz de 2 dimensões no caso de imagens em tons de cinza ou uma matriz de 3 dimensões, cada dimensão representando as intensidades dos pixels em cada canal de cor (RGB).

A arquitetura desse tipo de rede remonta ao modelo *Neocognitron*, apresentado por *Fukushima e Miyake* [74]. Essas redes têm como componente central as camadas convolucionais, que operam sobre a saída de camadas anteriores ou diretamente sobre uma entrada da rede, realizando uma operação de convolução discreta. Cada camada convolucional executa uma sequência de operações não lineares sobre uma região da entrada de cada vez, dada pelo tamanho do campo receptivo pré-definido (usualmente regiões quadradas de 3x3 pixels).

Cada camada convolucional executa um número pré-definido de convoluções sobre a sua entrada. Esse número é dado pela quantidade de filtros que a camada possui. Assim, para cada filtro, uma convolução é realizada sobre toda a imagem, em porções dela de cada vez, com tamanho dado pelo campo receptivo. O resultado das convoluções é um conjunto de mapas de características, cada um correspondente à convolução realizada por um dos filtros sobre a imagem. Os filtros armazenam os parâmetros a ser aprendidos, ou seja, os pesos de cada camada, e o tamanho de cada filtro é correspondente ao tamanho do campo receptivo (altura x largura) e à profundidade do bloco de dados de entrada (Figura 2.4).

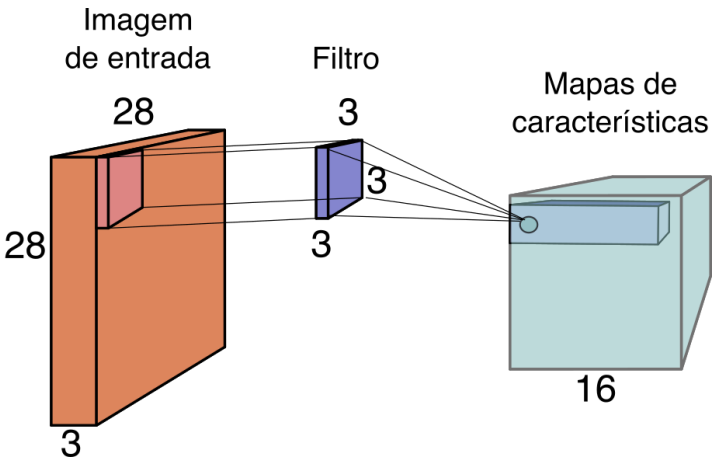


Figura 2.4: Esquemático da operação de uma camada convolucional. Cada filtro possui tamanho 3 x 3 x 3, correspondendo a um total de 27 parâmetros, somados a um parâmetro de viés. Os neurônios correspondem aos círculos, presentes nos mapas de características. O bloco de saída, com 16 camadas, corresponde a um exemplo em que houve a aplicação de 16 filtros à imagem de entrada.

Assim como ocorre com as redes neurais clássicas, os neurônios, ou unidades computacionais de cada camada, correspondem ao produto cartesiano entre os parâmetros e os valores de entrada, seguidos da aplicação da função de ativação, conforme definido pelas Equações 2.3 e 2.4. A diferença primordial entre os neurônios presentes em uma CNN e aqueles utilizados em RNCs é que no caso das redes convolucionais, cada neurônio é conectado a uma pequena região específica dos dados de entrada, enquanto no último os neurônios são conectados a todas as unidades de entrada da camada.

Essa configuração da camada convolucional faz com que os pesos presentes em cada filtro sejam compartilhados por todas as unidades existentes em um mesmo mapa de características, o que reduz drasticamente o número total deles. Adicionalmente, o sequenciamento de filtros permite que características diferentes e mais simples sejam combinadas para a formação de um conjunto de características mais complexas, de maneira que cada

característica possa ser reconhecida em diferentes partes da imagem de entrada, o que traz grande capacidade de representação.

Uma CNN usualmente possui um conjunto de camadas convolucionais, combinadas com outros tipos de camadas, como *pooling*, camadas de regularização como *dropout*, entre diversas outras possíveis. Essa construção permite que a rede, ao final, possa construir características com complexidade e abstração crescente ao longo das camadas, ao mesmo tempo que reduz ruídos inerentes a variações de cor, posição e tamanho relativo dos itens que compõem a cena.

Apesar das redes convolucionais terem capacidade de realizar todo o processo de extração de características e de apresentarem resultados a problemas de classificação, detecção e segmentação, algumas soluções as utilizam apenas na fase de extração de características. A classificação propriamente dita pode ser executada por outras ferramentas, como é o caso de modelos SVM. Os trabalhos desenvolvidos em [75, 76, 77] seguem essa abordagem.

Em [75] características como textura e distribuição espacial de núcleos celulares em imagens de biópsia em nódulos mamários são obtidas através de técnicas clássicas de processamento de imagem, que são combinadas com características de alto nível obtidas através de uma CNN. Essas características são utilizadas para o treinamento de um conjunto de SVMs com o objetivo de se identificar os graus de tecidos de câncer de mama.

Um trabalho semelhante foi desenvolvido por *Xu et al.* em [76, 77], utilizando uma CNN para a extração de características posteriormente utilizadas para a classificação e segmentação aplicadas ao contexto de diagnósticos de tumor cerebral. Nesse trabalho, os autores se utilizam de uma rede treinada sobre a base de dados ImageNet [16] para o processo de extração de características, em razão da baixa disponibilidade de imagens médicas para treinamento. Os autores destacam ainda como as características extraídas pela CNN são capazes de reconhecer propriedades de alto nível relevantes, como heteromorfismo celular, hemorragia celular, angiogenese e uniformidade de tamanho entre células nos tecidos analisados.

Assim, a alta capacidade que as CNNs possuem para produzir *features* de qualidade, bem como sua resistência a ruídos inerentes às diferentes tecnologias de digitalização de *slides*, as torna ideais para a aplicação em soluções de aprendizado ativo, especialmente para o trabalho aqui desenvolvido e detalhado no Capítulo 6. Essa capacidade inerente de extração de *features* pode ser combinada ainda a técnicas como *data augmentation* (abordado a seguir), com potencial de ganhos de efetividade adicionais.

2.2.2 *Data Augmentation*

Data Augmentation (DA) se refere a conjuntos de técnicas ou operações que podem ser efetuadas sobre imagens, aumentando a variabilidade de exemplos, para que um conjunto

de imagens usualmente pequeno ou insuficiente para o treinamento de um classificador possa ser usado com sucesso [78]. Os objetivos mais comuns ao se aplicar técnicas de DA são evitar o fenômeno do *overfitting* ou produzir melhores resultados de classificação com um mesmo conjunto de imagens.

Além dos objetivos já descritos, técnicas de *data augmentation* poderiam ser usadas para expandir um conjunto de treinamento inicial e de pequeno tamanho, gerando novas imagens a partir daquelas já anotadas. Todavia, a aplicação da técnica nesse contexto se enquadraria em um tipo de geração de bases de dados sintéticas, trazendo consigo os problemas inerentes a esse modelo de geração (referir ao Captítulo 3). Vale destacar que um dos principais problemas, intrínseco ao uso de DA com esse fim, é a representatividade que as imagens geradas possuem em relação ao universo de imagens da aplicação, especialmente considerando o uso de operações de processamento de imagens tradicionais. Um modelo treinado com imagens pouco representativas terá dificuldades de generalizar e classificar corretamente imagens não abrangidas por aquelas presentes no conjunto de treinamento.

Utilizar DA em conjunto com CNNs e aprendizado ativo, por outro lado, é plenamente possível e, em tese, benéfico em dois aspectos. Considerando que o AL tem como um de seus principais objetivos produzir conjuntos de treinamento mínimos e sabendo que a necessidade de volume de dados é inerente ao treinamento de CNNs, *data augmentation* poderia tanto beneficiar o treinamento, evitando a ocorrência de *overfit*, quanto permitir que níveis desejados de desempenho de classificação sejam atingidos com número menor de imagens.

Capítulo 3

Aprendizado Ativo

Obter dados de treinamento em quantidade suficiente para que as redes modernas atinjam desempenho satisfatório é um desafio reincidente em soluções de *deep learning* [14]. Enquanto algumas aplicações desfrutam de bases de dados extensas, construídas ao longo dos anos, outras acabam sendo inviabilizadas devido à falta de dados. As aplicações médicas são um dos exemplos mais emblemáticos da dificuldade em se construir bases de dados de tamanho e qualidade suficientes.

A obtenção de dados de treinamento anotados manualmente por voluntários é uma solução comumente adotada em outros domínios mas a especificidade dos problemas em medicina demanda que essas anotações sejam feitas por indivíduos especializados [15, 17, 18, 79, 80, 81]. Apesar do uso de voluntários permitir a obtenção de grande quantidade de dados de treinamento com custo reduzido, é necessário um esforço para a organização e disponibilização dos dados de maneira massificada, além da manutenção de processos de qualidade para garantir anotações precisas. Nesse caso, as discordâncias entre anotações devem receber especial atenção e devem ser analisadas por especialistas experientes, cujo tempo de trabalho usualmente possui alto custo.

A obtenção de bases de treinamento através da geração de dados sintéticos também é uma abordagem possível e já utilizada em alguns trabalhos [19, 20, 82, 83]. *Houl et al.*, por exemplo, propuseram um fluxo de geração de dados sintéticos usados para treinar modelos de segmentação de núcleos em WSIs. Já *Bug et al.* [82] apresentam uma técnica de *Image Quilting* [84] adaptada para o contexto das imagens histopatológicas, buscando gerar novos patches a partir de outros já existentes.

O uso desse tipo de dado reduz a dependência de anotações manuais, todavia, o desenvolvimento de métodos para produção de dados sintéticos para diferentes aplicações é uma área de pesquisa ainda muito ativa. Além das dificuldades em se gerar imagens conceitualmente válidas, é importante avaliar a representatividade dessas imagens dentro do universo a ser considerado. Adicionalmente, as técnicas de geração de imagens sintéticas

não têm foco em otimizar o processo de treinamento com a geração de imagens significativas, usualmente produzindo extensos bancos de dados, o que se reflete em maiores custos de treinamento das redes.

O aprendizado ativo se apresenta como uma abordagem para geração de bases de dados de treinamento de tamanho reduzido mas que possuam exemplos suficientes para que um modelo seja treinado e possa produzir um determinado patamar de desempenho aceitável. Basicamente, o AL é um processo iterativo em que um conjunto de treinamento é gradualmente expandido utilizando exemplos escolhidos por algum mecanismo de seleção, ao mesmo tempo em que o número total de exemplos contidos no conjunto deve ser mantido o menor possível. Os exemplos selecionados a cada iteração são apresentados a um oráculo externo para que sejam rotulados, de forma que possam ser usados no treinamento do modelo na iteração seguinte.

A dinâmica básica do aprendizado ativo elimina ou reduz drasticamente os problemas apresentados pelas outras técnicas de geração de bases de dados, uma vez que não é necessário um grande grupo de anotadores, podendo ser mantido um conjunto de pessoas coeso e experiente, limitando os problemas gerados por discordâncias de anotações. Essa característica também evita os custos e a estrutura de revisão demandados quando as anotações são feitas por muitos indivíduos. Como as imagens utilizadas são reais não existe também a preocupação de avaliar se imagens sintéticas de fato representam o universo estudado ou se foram inseridas características estranhas ao domínio.

A aplicação de AL no aprendizado de máquina já é estudada há décadas, inclusive com especializações dedicadas ao treinamento de redes neurais clássicas [22, 85, 86]. É importante salientar que não existe uma técnica de AL universalmente eficiente [87] e cada problema, composto de um universo de particularidades, pode apresentar uma ou mais técnicas com resultados satisfatórios. Outras podem apresentar desempenho inadequado, de forma que durante a criação de modelos de AL para um problema, podem ser usados como referência soluções já testadas em outros cenários mas não necessariamente essas soluções serão satisfatórias.

Nas seções seguintes serão apresentadas abordagens e propostas de AL para diferentes contextos do problema de classificação, considerando a disponibilidade de dados, o uso de modelos de aprendizado clássicos bem como o uso de modelos de *deep learning*.

3.1 Abordagens Quanto à Disponibilidade de Dados

Existem diferentes abordagens para o desenvolvimento de uma solução de AL, considerando a disponibilidade de exemplos não rotulados: consulta de membro (*membership query*), seleção a partir de fluxo (*stream-based selective sampling*) e seleção a partir de

estoque de dados (*pool-based selection*). Essas abordagens são apresentadas de maneira sucinta em um *survey* da área de aprendizado ativo [88].

A estratégia de AL por consulta de membro normalmente envolve um mecanismo de síntese de exemplos, que deverão ser rotulados, identificando se esses exemplos pertencem ou não a um conjunto previamente definido. Essa estratégia engloba aspectos do aprendizado ativo, como a consulta a um oráculo, mas é uma forma geral de produção de dados sintéticos que passam por um critério de seleção, desempenhado pelo oráculo, que definirá se os exemplos são parte do escopo da aplicação. Esse tipo de abordagem é de difícil implementação em casos em que o oráculo é um *expert* humano e especialmente se o universo de entrada corresponde a imagens [89].

Na abordagem por fluxo, os exemplos são apresentados ao modelo de aprendizado sequencialmente, e este decide se cada exemplo deve ser rotulado ou não. As estratégias usadas para a tomada de decisão usualmente se baseiam em medidas de incerteza ou do grau de informação contido em cada exemplo. Nesse caso, é necessário que os modelos de aprendizado sejam capazes de trabalhar com um exemplo de cada vez, incorporando-os ao conjunto de pontos rotulados de forma que, a cada novo exemplo, o aprendizado possa ser beneficiado com as informações adicionadas em uma nova rodada de treinamento.

Entre as abordagens apresentadas, o aprendizado ativo baseado em um estoque de dados (*pools*) é a mais utilizada, especialmente em aplicações que já possuem vastas bases de dados disponíveis, sendo aplicada a diferentes ferramentas de aprendizado. No caso das redes neurais convolucionais [88], essa abordagem se mostra a mais adequada, visto que as redes não trabalham bem com exemplos individuais, dependendo de grupos de exemplos de entrada de cada vez (*batches*). Adicionalmente, à medida que os conjuntos de treinamento vão crescendo, a adição de novos elementos possui efeito limitado no ganho de aprendizado, de forma que se torna inviável a seleção de unidades de cada vez.

Um *pool* é um conjunto de exemplos não rotulados, disponíveis para serem utilizados em um processo de treinamento após receberem rótulos. A abordagem de AL baseada em *pools* presume a existência de um conjunto desse tipo, composto de grande número de exemplos, do qual se deseja produzir rótulos a um subconjunto deles. Assim, ao conjunto de dados não rotulados U^0, é necessário extrair um conjunto de treinamento inicial L^0, para o qual rótulos devem ser produzidos. A cada iteração i do processo, um modelo M é treinado utilizando os pontos contidos no conjunto rotulado L^i e um novo subconjunto de pontos A^i, contendo s exemplos, é selecionado de U^i para ser incluído no subconjunto de treinamento da próxima iteração L^{i+1}, utilizando para isso uma função de aquisição $A^i = a(U^i, M)$.

Os pontos contidos em A^i devem ser rotulados por um oráculo externo e o processo se repete até um certo número pré-definido de iterações ou até que um determinado nível

de desempenho seja obtido. A partir da primeira iteração ($i = 0$), os conjuntos U e L são atualizados ao final de cada uma delas, conforme: $L^{i+1} = A^i \cup L^i$ and $U^{i+1} = U^i - A^i$.

Um aspecto fundamental em qualquer solução de AL é o mecanismo utilizado para implementar a função de aquisição $a(U^i, M)$. Essas funções podem ter como base diferentes abordagens matemáticas para selecionar os exemplos não rotulados que potencialmente apresentam a maior capacidade informativa para o aprendizado. A seguir são apresentados os tipos de funções mais utilizados para a abordagem baseada em estoque de dados.

3.2 Funções de Aquisição para Abordagem de Estoque de Dados

Algumas das estratégias de seleção de pontos para aprendizado ativo mais comumente encontradas na literatura, considerando a disponibilidade de um *pool* de dados, são: avaliação de incertezas [90, 91], avaliações geométricas ou por densidade de informação [92], bem como soluções híbridas, combinando incertezas e avaliações geométricas [93, 94, 95, 96].

Essas técnicas são apresentadas de maneira geral a seguir, como uma revisão histórica não exaustiva, com foco nas métricas que serão utilizadas no decorrer do presente trabalho.

3.2.1 Avaliações por Incertezas

Uma classe de funções de aquisição comumente utilizada é a de avaliações por incertezas. Nesse caso, o objetivo é identificar o candidato em que o modelo produziu uma previsão com o menor nível de confiança. Esse tipo de função utiliza métricas de incerteza que acabam por identificar os exemplos não rotulados que se encontram mais próximos dos limites de decisão dos modelos, com a espectativa de que seu rotulamento e posterior reconhecimento durante o treinamento produza uma aceleração do aprendizado.

O uso de métricas de incerteza como a única ferramenta de seleção de exemplos para anotação, todavia, demanda prévio conhecimento acerca das características dos dados sob análise, uma vez que essas métricas consideram que os elementos não rotulados são independentes e com distribuição idêntica (*independent and identically distributed - IID*) [97]. A utilização de uma função de aquisição, que se baseie apenas em incertezas, sobre um conjuntos de dados em que há uma correlação entre os exemplos pode produzir conjuntos de treinamento não ideais, com a presença de redundâncias que não contribuirão com o treinamento.

A seguir serão detalhadas métricas de incertezas comumente utilizadas na literatura.

Entropia

Uma das métricas de incertezas de maior destaque é a entropia máxima, derivada do trabalho desenvolvido por *Shannon* [98] no contexto das telecomunicações, como uma medida de quanta informação poderia ser extraída de um processo de Markov. A entropia do sistema $\mathbb{H}$ pode ser representada de maneira genérica como:

$$\mathbb{H} \;=\; -K \sum_{i=1}^{C} p_i log\ p_i, \tag{3.1}$$

em que p_i corresponde à probabilidade de ocorrência do evento i, C é o número de eventos possíveis e K é uma constante. As equações com essa forma apresentam as seguintes propriedades:

1. $\mathbb{H} = 0$ se e somente se todas as probabilidades p_i forem zero, à exceção de uma delas, à qual necessariamente deve ser 1,0. Ou seja, a entropia desaparece se há certeza absoluta sobre o evento observado;
2. Para um dado valor de C, caso todas as probabilidades sejam iguais ($p_i = \frac{1}{C}$), a entropia é máxima e equivalente a $log\ C$, correspondendo à situação de maior incerteza;
3. Considerando um evento conjunto correspondente à ocorrência de dois eventos distintos i, j, a entropia do sistema será dada pela probabilidade conjunta $p_{i,j}$:

$$\mathbb{H} \;=\; -K \sum_{i,j} p_{i,j} log\ p_{i,j}, \tag{3.2}$$

Para modelos probabilísticos, a entropia do sistema pode ser dada pelas probabilidades condicionais do modelo produzir um evento c, dado um exemplo de entrada x, levando em consideração os parâmetros atuais desse modelo. Assim, a entropia pode ser expressada como [99, 100]:

$$\mathbb{H}[y|x, \mathcal{D}_{train}] \;=\; -\sum_{c}^{C} p(y = c|x, \mathcal{D}_{train}) log\ p(y = c|x, \mathcal{D}_{train}), \tag{3.3}$$

em que $\mathcal{D}_{train}$ representa o conjunto de treinamento (X,Y) sobre o qual foi realizado o aprendizado do modelo. Nesse caso, o exemplo ou conjunto de exemplos com a maior entropia representaria os pontos em que o modelo apresenta as maiores incertezas de classificação, sendo aqueles selecionados para receberem rótulos.

VR

Outra métrica de incertezas baseada na baixa confiança de classificação dos modelos é *variation ratios* [101]. Essa medida se baseia na identificação da dispersão da previsão ao redor de um ponto central da distribuição de previsões geradas pelo modelo. No caso do problema de classificação, o ponto central adotado é a moda dos rótulos produzidos durante a previsão. Assim, *variation ratios* é definida de maneira genérica pela equação 3.4,

$$\begin{aligned} VR(x) &= 1 - \frac{f_x}{T}, \ dado\ que \\ f_x &= \sum_{t}^{T} \mathbb{1}[y^t = c^*] \end{aligned} \tag{3.4}$$

em que $\mathbb{1}[]$ corresponde à função indicador, $c^* = \underset{c=1,...,C}{\operatorname{argmax}} \sum_{t}^{T} \mathbb{1}[y^t = c]$ define a moda das previsões, f_x corresponde ao número de vezes em que a moda é o rótulo produzido pelo modelo quando T previsões são realizadas sobre o mesmo ponto de entrada x.

A partir de sua definição, é possível observar que o cálculo de incertezas através de *variation ratios* depende de modelos de previsão com comportamento estocástico, uma vez que modelos determinísticos, ao serem apresentados a um item de entrada múltiplas vezes, produzirão sempre a mesma saída, fazendo com que f_x seja igual a T, o que resulta em um nível de incerteza nulo.

O VR não é indicado para situações em que há o desbalanceamento de classes entre os exemplos não rotulados, pois esse desbalanceamento gera uma tendência de que as previsões produzidas sobre cada ponto x sejam, em grande número, da classe dominante. Dessa forma, a moda c^* tende a ser essa classe e $f_x \to T$, levando a distorções da incerteza produzida, que será baixa para a maioria dos pontos de entrada.

Bayesian Active Learning by Disagreement (BALD)

A terceira métrica de incertezas abordada nesse trabalho é *BALD* [102]. BALD procura avaliar a diferença entre dois tipos de incertezas: a incerteza preditiva e a incerteza do modelo. A incerteza preditiva é baseada nos valores médios da probabilidade que um ponto tem de pertencer a uma dada classe, enquanto a incerteza do modelo corresponde a variações na previsão produzida pelo modelo sobre um mesmo ponto.

Como exemplo do comportamento dessas categorias de incertezas, considerando um problema de classificação binária e os valores de probabilidades indicados para cada classe em uma previsão sendo $(c1; c2)$, a **incerteza preditiva** é alta se:

1. múltiplas previsões sobre um mesmo ponto obtenham resultados iguais a 0,5 para ambas as classes ($\{(0,5;0,5),...,(0,5;0,5)\}$), correspondendo ao cenário de maior entropia; e/ou

2. a metade das previsões tem probabilidade 1 e a outra tem probabilidade 0 ($\{(1,0),....,(0,1)\}$), indicando também alta entropia.

Por outro lado, há uma alta **confiança preditiva** em casos em que a previsão sobre esse ponto seja sempre 1 para uma dada classe ($\{(1,0),(1,0),(1,0),....,(1,0)\}$). Com relação à incerteza do modelo, a ocorrência de múltiplas previsões sucessivas e iguais para um mesmo ponto, mesmo com probabilidades (0.5,0.5) para as classes, indica uma alta confiança do modelo enquanto variações da previsão indicam incerteza do modelo com relação a esse ponto.

BALD define a incerteza preditiva como sendo a entropia do conjunto de treinamento e utiliza a probabilidade *a posteriori* sobre os parâmetros do classificador como a incerteza de modelo. Assim, a métrica BALD, aplicada a modelos estocásticos, pode ser expressa como:

$$\begin{aligned}\mathbb{I}[y,w|x,\mathcal{D}_{train}] &:= \mathbb{H}[y|x,\mathcal{D}_{train}] - \mathbb{E}_{p(w|\mathcal{D}_{train})}\left[\mathbb{H}[y|x,\omega]\right] \\ &= -\sum_c p(y=c|x,\mathcal{D}_{train}) log\ p(y=c|x,\mathcal{D}_{train}) \\ &+ \mathbb{E}_{p(w|\mathcal{D}_{train})}\left[\sum_c p(y=c|x,\omega) log\ p(y=c|x,\omega)\right]\end{aligned} \tag{3.5}$$

3.2.2 Avaliação Geométrica ou por Densidade de Informação

As métricas por avaliação geométrica recebem essa denominação por procurar correlações entre os exemplos de um conjunto de dados, de forma a identificar a similaridade ou dissimilaridade entre eles. Assim, é possível estabelecer uma medida de distância entre os exemplos, de forma que grupos possam ser identificados em meio ao conjunto total. A correlação entre um exemplo x_i e um conjunto de dados U^t (como o conjunto de pontos não rotulados) pode ser dada de maneira genérica como:

$$\mathcal{Q}(x_i) = \frac{1}{|U^t|}\sum_{x_j \in U^t, i \neq j} q(x_i, x_j), \tag{3.6}$$

onde $q(x_i, x_j)$ representa uma métrica de correlação entre os elementos i, j. A Equação 3.6 permite inferir que os pontos em que $\mathcal{Q}$ é alto indica uma alta correlação entre esse ponto e os demais do conjunto, de modo que esse ponto possui alta capacidade representativa sobre o universo de pontos em consideração. Os pontos com baixa correlação, por sua vez, são aqueles na borda do espaço de representação, usualmente considerados pontos isolados ou *outliers*.

Assim, funções de aquisição que se baseiam nesse tipo de métrica podem selecionar os exemplos com maior nível de correlação a um conjunto de dados por completo, com o intuito de se utilizar pontos representativos. Outra possível abordagem seria a subdivisão de um conjunto em grupos menores de dados e selecionar os itens com maior correlação dentro de cada subgrupo. *Tong & Koller* [103] adotam uma visão semelhante, ao definir um espaço de versões (*version space*) e, a partir dele, escolher o item não rotulado que melhor representa aqueles já rotulados (seleção *SimpleMargin*) ou o item que estabelece a melhor separação entre dois subconjuntos do espaço de versões (seleções *MaxMin* e *Ratio Margin*).

3.2.3 Funções Híbridas

A combinação de métricas de incertezas com métricas de avaliação geométrica forma uma função híbrida que pondera quanta influência uma ou outra abordagem deve ter na seleção dos pontos que deverão ser rotulados e adicionados ao conjunto de treinamento na iteração seguinte do aprendizado. Basicamente, o que se busca equilibrar são o grau de informação adicionado por cada novo exemplo (modelado pelo nível de incerteza associado aos exemplos) com a representatividade que esses exemplos apresentam sobre o todo.

Assim, no caso das funções de aquisição híbridas, decisões devem ser tomadas a respeito de quais métricas utilizar, bem como a forma em que elas devem ser combinadas para produzir uma lógica de seleção para os novos exemplos. *Li et al.* [94], por exemplo, propõem a utilização de uma relação multiplicativa da forma:

$$h_\beta(x_i) \;=\; f(x_i)^\beta d(x_i)^{1-\beta}, \tag{3.7}$$

em que β é um parâmetro do sistema, $f(x_i)$ representa uma métrica de incertezas e $d(x_i)$ é uma métrica de densidade. Nesse caso, o parâmetro β pode favorecer as métricas de incerteza caso seu valor seja maior que 0,5 ou favorecer as métricas de densidade caso seja menor que 0,5.

Outras funções de combinação certamente são possíveis, com atenção ao domínio em que a solução de AL deve ser implantada.

3.3 Processo de Aprendizado Ativo

Os conceitos apresentados até aqui nesse capítulo são necessários para se produzir uma definição formal do processo do aprendizado ativo. Esse processo é o resultado da escolha e combinação de múltiplos subprocessos, como cálculos de incertezas ou outras métricas utilizadas na seleção dos pontos a serem rotulados.

O processo de aprendizado ativo pode ser definido de maneira genérica como uma sequência de passos pré-definidos, conforme explicitado pelo Algoritmo 1. Nesse caso, é importante destacar que o tipo e número de métricas que podem ser utilizadas para se avaliar os exemplos não rotulados é variável. Com relação aos tipos, as métricas podem ser divididas entre aquelas que levam em consideração o modelo atual, o conjunto de exemplos não rotulados, o conjunto de exemplos já rotulados ou outros dados, conforme as peculiaridades de cada tipo.

O processo de combinação dessas métricas individuais é a etapa fundamental em que a função de aquisição irá definir uma métrica global a ser usada na seleção dos exemplos que deverão receber rótulos. Há soluções que usam apenas um tipo de métrica, em que a combinação é trivial, e as soluções híbridas, com múltiplas métricas avaliativas.

Algoritmo 1: Aprendizado ativo, procedimento geral.

Entrada: U^0 pool de patches não rotulados, Q número de exemplos a serem adquiridos; Z iterações de seleção/aprendizado
Resultado: L^z pontos selecionados; U^z

repita
  $\mathcal{M}^{t+1} \leftarrow$ `TreinaModelo(`$\mathcal{M}^t, L^t$`)`;
  (PA) $\leftarrow$ `MetricaA(`U`,...)`;
  (PB) $\leftarrow$ `MetricaB(`U`,...)`;
  .....;
  (PM) $\leftarrow$ `MetricaM(`U`,...)`;
  (CM) $\leftarrow$ `CombinarMetricas(Ordenar(`PA`),...,Ordenar(`PM`))`;
  A $\leftarrow$ `ObterPontos(`CM`,`Q`)`;
  $U^{t+1} \leftarrow U^t - A$;
  $L^{t+1} \leftarrow L^t + A$
até *Z iterações ou nível de performance*;

Assim, após um número máximo de iterações Z ou após se atingir um nível de desempenho aceitável, o processo é finalizado, com a obtenção de um conjunto de treinamento rotulado contendo $Z * Q + |L^0|$ exemplos.

3.4 Técnicas Aplicadas a Modelos de Aprendizado Clássico

Apesar de demandarem bases de treinamento significativamente menores que modelos de *deep learning*, como as CNNs, modelos clássicos também podem demandar esforços de anotação muito altos e já foram alvo de soluções de aprendizado ativo.

O trabalho desenvolvido por *Li & Guo* [94] é um exemplo importante aplicado a classificadores probabilísticos, sendo utilizados **regressores logísticos**. A solução apresentada

é um híbrido que envolve o uso de incertezas (entropia máxima) e uma medida de densidade, desenvolvida pelos autores. O trabalho merece destaque por apresentar claramente todas as etapas de construção de uma solução de aprendizado ativo, com a definição de métricas, como elas podem ser combinadas e a seleção de exemplos propriamente dita. Essa estrutura básica em que uma métrica de incerteza é combinada com uma métrica de densidade foi usada como inspiração para o trabalho ora desenvolvido.

A métrica de densidade criada pelos autores é desenvolvida sobre um Processo Gaussiano para o cálculo das entropias conforme a equação 3.8:

$$\begin{aligned} d(x_i) &= H(x_i) - H(x_i|X_{U_i}),\ e \\ H(x_i) &= \frac{1}{2}\ ln(2\pi e\sigma_i^2), \\ H(x_i|X_{U_i}) &= \frac{1}{2}\ ln(2\pi e\sigma_{i|U_i}^2), \\ d(x_i) &= \frac{1}{2} ln\left(2\pi e\sigma_{i|U_i}^2\right), \end{aligned} \tag{3.8}$$

onde $\sigma_i^2 = \mathcal{K}(x_i, x_j) = exp\left(-\frac{(||x_i - x_j||^2}{\tau^2}\right)$ corresponde ao *kernel* Gaussiano e $\sigma_{i|U_i}^2$ é obtido a partir de uma matriz de covariância (referir a [94] para detalhes).

A métrica de incerteza $f(x_i)$ é combinada com a densidade através de uma relação de troca simples regulada por um parâmetro β, em que $h_\beta(x_i) = f(x_i)^\beta d(x_i)^{1-\beta}$. Todavia, a escolha do parâmetro β não é trivial, visto que o comportamento da relação de troca não é estático. Para lidar com essa escolha, os autores apresentaram uma técnica inicial, em que o melhor valor dessa variável é reavaliado periodicamente e escolhido a partir de um conjunto pré-definido de valores.

Em outra abordagem, o uso de agrupamentos (ou *clusters*) para decompor os itens de um conjunto de amostras no contexto do aprendizado ativo já foi proposto anteriormente por *Nguyen & Smeulders* [104]. Em [104], uma proposta de formação de *clusters* para o problema de classificação binária foi formalmente incorporada em uma solução de aprendizado ativo. A solução é demonstrada utilizando um classificador SVM para o problema de reconhecimento facial.

No cenário tratado pelos autores, a criação de agrupamentos é utilizada para aproximar a distribuição *a priori* dos dados, na hipótese de que membros do mesmo grupo têm alta probabilidade de possuírem o mesmo rótulo. Como será demonstrado pelos experimentos descritos no Capítulo 7, essa hipótese não é razoável para o caso de bancos de dados com elevado desbalanceamento de classes. São contribuições importantes do trabalho desenvolvido em [104] a formalização da distribuição dos dados em *clusters*, bem como a determinação de como os rótulos dos pontos representativos do *cluster* podem ser disseminados para outros membros do conjunto.

De maneira assemelhada ao trabalho de *Li & Guo* [94] (que na verdade é anterior ao de *Nguyen & Smeulders* [104]), a estratégia para seleção de pontos adotada nesse último é definida a partir de uma combinação de uma métrica que procura identificar os pontos próximos da fronteira de classificação e uma métrica de densidade, que é derivada da avaliação de quanto compactos são os *clusters* formados pelo conjunto de pontos não rotulados.

Apesar das técnicas descritas serem aplicáveis a modelos clássicos de aprendizado, elas trazem inspirações para o desenvolvimento do modelo de aprendizado ativo que será apresentado no Capítulo 6.

3.5 Técnicas Aplicadas a Modelos de *Deep Learning*

As propostas de metodologias de AL aplicáveis a modelos de *deep learning*, mais especificamente aquelas desenvolvidas para CNNs, são relativamente recentes na literatura. Em grande parte, isso se deve ao fato de que as metodologias de AL disponíveis não escalavam bem para cenários em que o universo de dados possuía alta dimensionalidade [105]. Adicionalmente, as métricas populares de incertezas, utilizadas por muitas propostas de AL até então desenvolvidas, não poderiam ser diretamente aplicadas a esses modelos, uma vez que as CNNs possuem comportamento determinístico, ou seja, considerando um conjunto de parâmetros do modelo, sucessivas previsões sobre a mesma entrada produzem resultados idênticos. Assim, o uso de métricas consagradas na literatura dependia de um desenvolvimento teórico que permitisse seu uso junto a CNNs.

Esse desenvolvimento teórico, salvo melhor juízo, teve sua primeira aplicação prática produzida por Yarin Gal [100], abrindo possibilidades para novas modelagens e também para adaptações. A seção 3.5.1 apresentará um breve histórico de abordagens que buscaram produzir esse tipo de interpretação, aplicáveis a modelos de *deep learning*.

3.5.1 Modelagem Estocástica para Soluções de *Deep Learning*

De maneira genérica, o objetivo de uma modelagem estocástica para modelos de classificação é estimar uma função f^*, que aproxima uma função f real, e que provavelmente gerou as saídas observadas para um dado evento. Assim, seria possível prever qual seria a saída y^* da função f^* se a ela fosse fornecido o ponto de entrada x^*, até então não conhecido:

$$p(y^*|x^*, X, Y) \;=\; \int p(y^*|f^*)p(f^*|x^*, X, Y)df^*, \tag{3.9}$$

dado um conjunto de observações X e de rótulos Y possíveis para os dados, em relação às funções f^* que poderiam explicar os resultados observados.

Considerando que um modelo de aprendizado utilizado para classificação é uma função que busca representar a relação entre um conjunto de pontos de entrada e seus rótulos e que essa função f^* é definida por seus parâmetros ω (como os pesos de camada convolucionais), a equação 3.9 pode ser reescrita como:

$$p(y^*|x^*, X, Y) \;=\; \int p(y^*|f^*)p(f^*|x^*, \omega)p(\omega|X, Y)df^*d\omega \tag{3.10}$$

As redes neurais bayesianas (*Bayesian Neural Networks - BNNs*) [106, 107, 108] são pioneiras na aplicação desse tipo de modelagem estocástica para redes neurais clássicas e inspiram a produção de soluções aplicáveis a modelos de *deep learning*. Na prática, as BNNs eram propostas através da aplicação de probabilidades *a priori* sobre seus parâmetros ω ($p(\omega)$) e o desenvolvimento de técnicas de aproximação das probabilidades *a posteriori* do modelo a partir dessa distribuição inicial. Assim, o que se busca é a **aproximação da probabilidade posterior** sobre os pesos ($p(\omega|X, Y)$), que normalmente não se pode avaliar analiticamente.

As abordagens iniciais buscavam aproximar a probabilidade *a posteriori* dos modelos com o uso de inferência variacional (*variational inference* - VI) usando uma Gaussiana. A aproximação é obtida pela minimização da divergência de Kullback-Leibler (KL) entre uma distribuição variacional $q_\theta(\omega)$ e a probabilidade posterior, que, para uma função específica f^ω, é dada pela equação 3.11 [100]:

$$\begin{aligned} KL(q_\theta(\omega)||p(\omega|X, Y)) &\approx -\int q_\theta(\omega) log\ p(Y|X, \omega)d\omega \;+\; KL(q_\theta(\omega)||p(\omega)) \\ &= -\sum_{i=1}^{N} \int\; q_\theta(\omega) log\ p(y_i|f^\omega(x_i))d\omega \;+\; KL(q_\theta(\omega)||p(\omega)), \end{aligned} \tag{3.11}$$

onde $f^\omega(x_i)$ pode corresponder à saída do modelo quando apresentado à entrada x_i e N corresponde ao número de exemplos na base de dados. Essa modelagem, apresentada por *Hinton & Van Camp* [106], foi demonstrada apenas para uma rede com uma única camada oculta, visto que o logaritmo de verossimilhança é intratável para grande parte dos modelos [100] e a técnica requer seu cálculo para cada exemplo de entrada.

Em trabalhos mais recentes, que aplicaram VI para a aproximação da probabilidade *a posteriori*, como de *Blundell et al.* [108], foi possível aplicar a técnica de VI sobre modelos mais complexos e obter resultados comparáveis aos demais modelos utilizados à época, resolvendo parte dos problemas que tornavam a técnica de VI impraticável, todavia adicionando grande número de parâmetros.

CNNs Bayesianas

As abordagens anteriormente demonstradas, empregadas para redes neurais clássicas, produziam um aumento do número de parâmetros do modelo, que no caso de CNNs e especialmente para a aplicação em estratégias de AL, seriam impraticáveis do ponto de vista do custo computacional. *Yarin Gal & Zoubin Ghahramani* [109] abordam essa questão através do uso de aproximações de *Bernoulli* para distribuições variacionais, ao invés de inferência variacional, eliminando a necessidade de novos parâmetros para o modelo. Sua formulação primeiramente é demonstrada em redes neurais clássicas e em seguida estendida para CNNs.

Essa nova formulação para a aproximação das probabilidades posteriores considera que o uso de técnicas de regularização, como *dropout* [110], pode ser interpretada como uma aproximação da integral presente na Equação 3.11, através da integração de Monte Carlo [99]. A regularização via *dropout* busca reduzir o *overfit* adicionando variáveis binárias a cada unidade da rede, permitindo que suas saídas sejam computadas durante o treinamento, com uma probabilidade p_l, configurável para cada camada. Assim, considerando uma dada camada l, seus pesos podem ser definidos como:

$$\begin{aligned} W_l &= M_l * Z_l, e \\ Z_l &= [z_{l,j}]_{j=1}^{K_{l-1}},\ l = 1, ..., L;\ e\ j = 1, ..., K_{i-1} \end{aligned} \tag{3.12}$$

em que M_l são os parâmetros a serem otimizados e Z_l é uma matriz que armazena as variáveis binárias $z_{l,j}$. Essas variáveis apresentam valor zero com probabilidade p_l para cada uma das K_{l-1} unidades da camada l. As variáveis binárias introduzidas pelo uso de *dropout* seguem uma distribuição de Bernoulli, dado que possuem valor 1 com probabilidade p_l e valor 0 com probabilidade $1 - p_l$. Partindo dessas considerações, a função objetivo a ser otimizada durante o treinamento de uma rede que utiliza *dropout* como regularização estocástica e L_2 como regularização de pesos é:

$$\mathcal{C}_{dropout} = \frac{1}{N}\sum_{i=1}^{N} E(y_i, \hat{y}_i) + \lambda \sum_{l=1}^{L}(||W_l||_2^2 + ||b_l||_2^2), \tag{3.13}$$

em que b_l é o vetor de parâmetros de viés e E(.,.) corresponde à função softmax quando o rótulo do exemplo x_i é y_i e a previsão produzida pela rede é $\hat{y}_i$ para cada exemplo i de um conjunto de N pontos. Em seu desenvolvimento teórico, os autores [100, 109, 111] demonstram a equivalência entre as funções objetivo representadas pelas Equações 3.11 e 3.13. Assim, a previsão produzida pelo modelo, dada pela Equação 3.10, pode ser aproximada conforme a Equação 3.14:

$$
\begin{aligned}
p(y^*|x^*, X, Y) &\approx \int p(y^*|x^*, \omega) q_\theta(\omega) d\omega, \ e \\
&\approx \frac{1}{T} \sum_{t=1}^{T} p(y^*|x^*, \hat{\omega}_t),
\end{aligned}
\tag{3.14}
$$

considerando que a probabilidade posterior $p(\omega|X, Y)$ pode ser aproximada pela distribuição gerada pela aplicação de *dropout* ($q_\theta(\omega)$) sobre as unidades da rede. Nesse caso, durante a fase de previsão, são executadas T passagens sobre cada exemplo, cada uma utilizando os pesos gerados pela aplicação de *dropout* no momento de inferência ($\hat{\omega}_t$) e considerando a aproximação $\hat{\omega}_t \sim q_\theta(\omega)$. A probabilidade final de que o modelo gere a previsão obtida é, então, a média da saída produzida pela função *softmax* ao final das T passagens. A técnica apresentada por *Yarin Gal & Zoubin Ghahramani* e brevemente aqui descrita é conhecida como **MC Dropout**.

O modelo teórico até então apresentado se aplica a redes neurais clássicas. Todavia, sua aplicação a CNNs é direta, bastando que as camadas de *dropout* sejam inseridas após cada camada convolucional e antes de camadas de *pooling* [109]. Nesse caso, é importante salientar que, caso uma rede CNN Bayesiana seja utilizada para a realização de previsões, a estratégia de *MC Dropout* deve ser utilizada também nessa fase, com a realização de T passagens sobre cada item do conjunto de testes para que uma previsão final seja obtida.

A necessidade de se utilizar *dropout* após cada camada convolucional foi analisada em um trabalho posterior, de *Zeng et al.* [112]. Nesse trabalho, os autores avaliaram tanto o número do que foi chamado de "camadas Bayesianas" bem como sua posição em uma CNN simples. As conclusões a que os autores chegaram sugerem que a presença de um pequeno número de camadas Bayesianas mais próximas da saída da rede seria suficiente para a realização da aproximação das probabilidades posteriores, com a integração de Monte Carlo desenvolvida por *Gal et al.*

CNN Ensembles

Ensembles são conjuntos de modelos de aprendizado combinados para gerar melhores resultados preditivos sobre um dado domínio. Um dos objetivos da formação de ensembles é introduzir diversidade ao modelo final, que, em teoria, possuiria mais flexibilidade e capacidade de representação, superando os resultados individuais de cada constituinte do ensemble [113]. Vale destacar a aplicação do teorema do júri de Condorcet, que define que em um problema de decisão binária, em que cada membro de um conjunto de votantes $v \in V$ vota de maneira independente e tem probabilidade p_v de estar correto, sendo L o resultado da decisão colegiada, então, se $p_v > 0.5 \ \forall \ v \in V \rightarrow L > p_v \ \forall \ v \in V$ [113].

A utilização de ensembles de classificadores em tarefas de aprendizado de máquina data de muitas décadas, com aplicações focadas em redes neurais clássicas desenvolvidas desde

os anos de 1990 [114]. Usualmente, a motivação para o uso de ensembles é melhorar o desempenho preditivo de um modelo. Todavia, o uso de ensembles para derivar incertezas sobre a previsão de modelos de redes neurais, quando expostas a exemplos ainda não vistos, já foi alvo de trabalhos anteriores [115, 116].

A aplicação de ensembles para a derivação de incertezas utilizando CNNs foi explorada por *Beluch et al.* [117]. A técnica de *MC Dropout*, apresentada anteriormente, pode ser interpretada como a geração de um membro de um ensemble a cada passagem sobre os dados (cada membro representado pelos parâmetros $\hat{w}_t$) e o cálculo da média ao final de T passagens produz o resultado equivalente ao de um ensemble cuja saída seja a média das previsões de seus membros [115].

A partir dessa visão, a geração de incertezas com o uso de ensembles segue a mesma modelagem estocástica utilizada na produção de incertezas via MC Dropout, mas as T passagens sobre os dados são substituídas pelas saídas de E membros do ensemble, conforme a Equação 3.15:

$$\begin{aligned} p(y^*|x^*, X, Y) &\approx \int p(y^*|x^*, \omega) q_\theta(\omega) d\omega, \ e \\ &\approx \frac{1}{E} \sum_{e=1}^{E} p(y^*|x^*, \hat{\omega}_e), \end{aligned} \tag{3.15}$$

em que $\hat{\omega}_e$ correspondem aos pesos finais (após o treinamento) de cada membro do ensemble. Nesse caso, todos os membros são treinados sobre o mesmo conjunto de dados, a partir de pesos com inicializações distintas.

Capítulo 4

Trabalhos Relacionados

O aprendizado ativo tem sido utilizado com diversas ferramentas de aprendizado de máquina ao longo das décadas. Originalmente, modelos de regressão logística e SVMs eram as mais comuns, sendo demonstradas aplicações com diferentes graus de sucesso. Com o renascimento de modelos de IA em tarefas de visão computacional, o foco foi alterado para modelos de *deep learning*, o que demandou o desenvolvimento de novas abordagens de AL.

Nas seções seguintes são apresentadas, como revisão bibliográfica, soluções de AL voltadas para ferramentas de *deep learning*. Para as soluções que demandam cálculos de incertezas, será demonstrado como o desenvolvimento teórico com interpretação estatística para CNNs é usado nesses cálculos. Adicionalmente, são abordadas técnicas de simplificação de CNNs já consolidadas na literatura, de forma a justificar o desenvolvimento do NAR nos capítulos seguintes.

4.1 Propostas de AL para Modelos de *Deep Learning*

Diferentes abordagens de AL foram propostas para melhorar o processo de geração de bases de dados de treinamento para CNNs (destacadamente [92, 118, 117]), demonstrando bons resultados sobre diversos problemas. Essas três abordagens são utilizadas como inspiração para o desenvolvimento do trabalho aqui proposto e são usadas como referências de comparação durante os experimentos conduzidos.

4.1.1 *Deep Bayesian Active Learning (MC Dropout)*

A abordagem de AL proposta por *Gal et al.* [118] utiliza funções de aquisição por incertezas, que são calculadas a partir do desenvolvimento teórico apresentado anteriormente para CNNs Bayesianas, e aplicadas sobre um *pool* de exemplos não rotulados. Assim, as

métricas de incertezas apresentadas na seção 3.2.1 podem ser utilizadas com o modelo estocástico desenvolvido, a partir de pequenas adaptações. Os elementos selecionados para receberem rótulos por um oráculo externo serão simplesmente aqueles A elementos que possuem os maiores valores obtidos por essas métricas.

O cálculo da entropia pode ser realizado conforme a Equação 4.1, extraindo-se os A elementos que apresentam os maiores valores:

$$\begin{aligned}
\mathbb{H}[y|x,\mathcal{D}_{train}] &= -\sum_{c}^{C} p(y=c|x,\mathcal{D}_{train}) log\ p(y=c|x,\mathcal{D}_{train}) \\
&= -\sum_{c}^{C} \int p(y=c|x,\omega)\ p(\omega|\mathcal{D}_{train}) d\omega\ *\ log \int p(y=c|x,\omega) p(\omega|\mathcal{D}_{train}) d\omega \\
&\approx -\sum_{c}^{C} \int p(y=c|x,\omega)\ q_\theta(\omega) d\omega\ * log \int p(y=c|x,\omega) q_\theta(\omega) d\omega \\
&\approx -\sum_{c}^{C} \left(\frac{1}{T} \sum_{t} \hat{p}_c^t \right) log \left(\frac{1}{T} \sum_{t} \hat{p}_c^t \right),
\end{aligned} \tag{4.1}$$

considerando que $\hat{p}_c^t = softmax(f^{\hat{\omega}_t}(x))$, corresponde à saída da função softmax para cada possível classe c que o exemplo de entrada x pode assumir, no contexto das aproximações desenvolvidas.

A métrica BALD, expressa pela Equação 3.5, pode ser obtida de maneira similar, com aproximação semelhante aplicada a seu segundo termo, resultando em:

$$\begin{aligned}
\mathbb{I}[y,w|x,\mathcal{D}_{train}] &\coloneqq \mathbb{H}[y|x,\mathcal{D}_{train}] - \mathbb{E}_{p(w|\mathcal{D}_{train})}\left[\mathbb{H}[y|x,\omega]\right] \\
&\approx -\sum_{c}^{C} \left(\frac{1}{T} \sum_{t} \hat{p}_c^t \right) log \left(\frac{1}{T} \sum_{t} \hat{p}_c^t \right) + \frac{1}{T} \sum_{c,t} \hat{p}_c^t log\ \hat{p}_c^t,
\end{aligned} \tag{4.2}$$

Uma consideração válida pode ser feita a respeito da métrica *variation ratios*, em relação à modelagem estocástica já apresentada. O cálculo da métrica apresentado na Equação 3.4 implica que deve haver variabilidade na classificação produzida pelo modelo sobre os itens do conjunto não rotulado, do contrário, $VR(x) = 0$. Esclarecida essa particularidade, a métrica VR pode ser obtida ao se considerar a aproximação da probabilidade de previsão do modelo sobre a classe moda (c^*), conforme a relação:

$$VR(x) = 1 - p(y=c^*|x,\mathcal{D}_{train}),$$

em que $\mathcal{D}_{train} = X, Y$ [100].

Em seu trabalho, *Gal et al.* [118] utilizam as métricas apresentadas, bem como uma seleção aleatória como *baseline*, sobre duas bases de dados. A primeira, popularmente utilizada em trabalhos de visão computacional, é a base MNIST [73], composta de dígitos

numéricos manuscritos, bem como uma base para aplicações de diagnósticos sobre câncer de pele.

Os experimentos sobre a base MNIST demonstraram a eficácia da estratégia de AL através da seleção de exemplos considerando as 3 métricas de incertezas já discutidas sobre o *baseline* da seleção aleatória. Nesse caso, a métrica *variation ratios* foi a que apresentou os melhores resultados. À época da publicação da abordagem de AL via MC Dropout, havia pouco material na literatura aplicado a CNNs, de forma que os autores utilizaram, como base de comparação em seus experimentos, abordagens que utilizam RBF Kernels [119] e SVMs para a seleção das imagens a serem rotuladas.

Entre os resultados obtidos sobre a base de dados real para diagnóstico de câncer de pele é importante destacar que, ao contrário do que se observou para os experimentos sobre a base MNIST, a métrica de incertezas que apresentou melhor resultado foi BALD.

4.1.2 *Ensembles for Active Learning*

Beluch et al. [117] aplicam a derivação de incertezas a partir de ensembles de CNNs conforme já descrito na seção 3.5.1 para calcular as métricas VR, BALD e Entropia Máxima, construindo funções de aquisição sobre essas métricas.

A estratégia via ensembles é demonstrada sobre três bases de dados, a saber: MNIST [73], CIFAR-10 [120] e uma base de dados de aplicação médica para classificação de retinopatia diabética. Nesse trabalho, os autores reportam níveis de AUC acima daqueles obtidos pelas abordagens Bayesiana de *Gal et al.* [118] e geométrica [92], esta última descrita em maiores detalhes a seguir.

Apesar de apresentar resultados que os colocam no estado da arte, os autores destacam uma das maiores dificuldades de aplicação dessa abordagem, que é o alto custo computacional para se treinar um conjunto de CNNs para compor o ensemble. Como uma solução para essa questão, os autores apontam a possibilidade de uso de técnicas de *implicit ensembling*, a partir de 3 abordagens: *snapshot ensembling* [121], *diversity encouraging ensemble (DEE)* [122] e *splithead ensemble* [123]. Todas as três técnicas apresentaram acurácia abaixo da aplicação de um ensemble completo utilizando a métrica de incertezas *variation ratios* aplicadas à base de dados CIFAR-10. Os autores, todavia, não apresentaram uma explicação para tal.

Os resultados obtidos sobre a base de dados MNIST e CIFAR-10 mostram a vantagem da abordagem por ensembles, sendo o melhor desempenho atingido com a utilização de VR como métrica de incerteza. Os autores, assim como foi feito no presente trabalho, utilizam a abordagem geométrica desenvolvida por *Sener & Savarese* [92] como referência, obtendo resultados modestos para a versão mais simples dessa abordagem, o que destaca a vantagem da estratégia por ensembles sobre a geométrica.

Em suas análises, *Beluch et al.* dedicaram maior atenção para a comparação entre os resultados obtidos entre as técnicas de ensembles e MC Dropout. Foram investigados os fatores que favorecem a utilização de ensembles, sendo obtidos resultados importantes para explicar os ganhos observados. Em suma, os principais parâmetros analisados e seus achados foram:

- Número de membros do ensemble: mesmo com 3 membros um ensemble obtém resultados superiores àqueles produzidos pela solução via MC Dropout com 100 passagens sobre os dados;
- Capacidade do modelo: o uso de *dropout* no momento de previsão reduz a capacidade do modelo, visto que uma parcela expressiva dos nós da rede é desabilitada. Experimentos realizados pelos autores sugerem que esse efeito possui impacto nos resultados obtidos pela abordagem de MC Dropout;
- Inicialização: a abordagem de MC Dropout pode ser interpretada como um ensemble em que o conjunto de passagens sobre os dados formam membros. Todavia, esses membros teriam inicializações de pesos iguais e podem convergir para um mesmo ponto ótimo local, limitando o resultado obtido. Esse fator revelou ter um impacto significativo nos experimentos realizados em [117];

Quando aplicado sobre a base de dados de retinopatia, a solução via ensembles, utilizando *variation ratios* como métrica de incertezas e InceptionV3 [124] como a CNN a ser treinada, a solução de AL obteve resultados melhores que uma seleção aleatória por uma ampla margem. Adicionalmente, um comportamento relevante da solução via ensembles foi observado, que foi a aquisição de um percentual expressivo de exemplos com rótulo positivo, quando comparado à aquisição aleatória.

4.1.3 *Geometric Approach for Active Learning*

Uma das abordagens de AL com maior relevância na literatura é aquela desenvolvida por *Sener & Savarese* [92], utilizando um modelo desenvolvido sobre o problema *core-set selection* para identificar os exemplos com maior representatividade dentro de uma base de dados. Essa abordagem é considerada uma forma de aplicação geométrica para AL ou por densidade de informação, como exposto por *Settles* [88].

Ao contrário das demais abordagens para AL, a via geométrica busca selecionar pontos para serem rotulados com base nas características da própria base de dados ao invés de incertezas produzidas pela ferramenta de classificação. Os autores argumentam que o uso de *batches* de exemplos para o treinamento/seleção de imagens em CNNs tem como efeito a criação de correlações entre as imagens selecionadas pelos algoritmos de AL, o

que tornaria as abordagens por incertezas inadequadas para a aplicação de aprendizado ativo sobre CNNs.

A abordagem geométrica, via modelagem sobre o problema *core-set*, visa identificar um subconjunto dos exemplos não rotulados que apresentariam um erro médio de classificação dentro de um limite, considerando os demais exemplos presentes no conjunto total. O objetivo do aprendizado ativo, nesse caso, seria identificar o subconjunto que minimiza esse limite, expresso formalmente através da Equação 4.3:

$$\min_{s^{k+1}:|s^{k+1}|\leq b} E_{x,y}[l(x,y;A_{s^0\cup\ldots s^{k+1}})], \tag{4.3}$$

em que s^i corresponde ao conjunto de exemplos selecionados para receberem rótulos na iteração i e $l(x,y;A_{s^0\cup\ldots s^{k+1}})$ corresponde ao erro obtido durante o treinamento quando a ferramenta de aprendizado A recebe como entrada o ponto x, selecionado aleatoriamente da base de dados, e seu rótulo y.

O desenvolvimento teórico apresentado mostra que identificar o conjunto s^{k+1} que minimiza o erro esperado, conforme a Equação 4.3, é equivalente a solucionar o problema de k-centro. Esse problema basicamente consiste em identificar os $b_i \in s^{k+1}$ pontos cuja maior distância entre eles e demais pontos vizinhos $v_i \notin s^{k+1}$ seja minimizada. Apesar desse problema ser NP-difícil, existem aproximações que apresentam bons resultados.

Importante destacar que, em [92], os autores apresentam uma versão gulosa do algoritmo que soluciona o problema k-centro, bem como uma versão que garante uma aproximação teórica com garantias formais mais restritas. Todavia, os experimentos demonstraram que a diferença entre ambas versões é pequena, de forma que são feitas comparações referentes à versão gulosa nos experimentos realizados e descritos no Capítulo 7.

Em seus experimentos, os autores obtiveram melhores resultados quando comparados a outros modelos, dos quais se destacam: MC-Dropout [118], referido em [92] como DBAL, k-Median, uma solução que seleciona para aquisição os pontos centrais identificados após particionar a base de dados em k clusters, e CEAL [125], abordado brevemente na Seção 4.1.4.

Adicionalmente, os autores conduziram uma análise de correlação entre os pontos inicialmente presentes no conjunto de treinamento, pontos selecionados para serem incluídos no conjunto em uma iteração seguinte e os demais ainda presentes no estoque. Os resultados demonstram que os pontos selecionados unicamente a partir de incertezas apresentam grande concentração em relação ao total de pontos disponíveis, enquanto aqueles selecionados pela abordagem geométrica cobrem o espaço de *features* de maneira mais ampla.

4.1.4 Outras Propostas de AL Aplicáveis a CNNs

Além das soluções já destacadas e descritas em pormenores nas seções anteriores, outras abordagens com resultados significativos estão disponíveis na literatura. Algumas delas são apresentadas brevemente a seguir.

O método CEAL [125] define uma proposta referida pelos autores como *cost-effective*, no sentido de que se busca maximizar o desempenho atingido pela rede a cada iteração, através do uso de exemplos a partir de duas formas de seleção: rotulagem manual daqueles com maior nível de incerteza e rotulagem automática dos exemplos que demonstram alta confiança de classificação do modelo. Com essa abordagem os autores pretendem criar grandes bancos de dados anotados, fornecendo exemplos de treinamento suficientes mesmo para as maiores redes.

Para esse fim, o método CEAL descreve 3 métricas de incertezas, que consideram a saída da função Softmax: $p(y_i = c|x_i, \omega) = softmax(f^{\omega}(x_i))$ como sendo a probabilidade do rótulo correto c de um exemplo x_i. As métricas apresentadas são *least confidence*, que ordena os exemplos não rotulados segundo o maior valor de probabilidade produzido pelo softmax, *margin sampling*, que considera a diferença de probabilidade entre os dois maiores valores do softmax, e a entropia, descrita conforme a Equação 4.1.

Os autores utilizam os exemplos rotulados automaticamente em um passo de treinamento da CNN, em conjunto com aqueles rotulados manualmente, de modo que a rede possa ser treinada com um conjunto expandido de exemplos. Todavia, os pontos com rótulos automáticos não são integrados ao conjunto daqueles que receberam rótulos manuais, sendo retornados ao conjunto não rotulado para a iteração seguinte do AL. Apesar dos autores demonstrarem ganhos sobre uma seleção aleatória, atingindo o desempenho máximo possível nas bases testadas após cerca de 60% dos exemplos rotulados, o CEAL é superado pela solução desenvolvida por *Sener & Savarese*.

Em um trabalho recente, *Shen et al.* [126] aplicam agrupamento antes da seleção de pontos utilizando CNNs. Em seu trabalho, os autores propõem uma função objetivo customizada para o algoritmo de clusterização K-Means. Essa nova função é baseada na proposta teórica desenvolvida por [92], que define que a diferença entre a perda média de treinamento em um conjunto de pontos S e um subconjunto T é limitada pelo raio $\delta_{T \to S}$, considerando que o conjunto T é um *core set* de S com o mesmo raio:

$$\left| \frac{1}{|S|} \sum_{x_i, y_i \in S} l(x_i, y_i; \theta_S) - \frac{1}{|T|} \sum_{x_i, y_i \in T} l(x_i, y_i; \theta_T) \right| \leq \mathcal{O}(\delta_{T \to S}) + \mathcal{O}\left(\frac{1}{\sqrt{|S|}} \right), \tag{4.4}$$

onde x_i, y_i são um exemplo e seu rótulo correspondente, l é a função de custo utilizada

pelo modelo, θ_S e θ_T são os parâmetros do modelo após este ter sido treinado sobre S e T respectivamente. Considerando T como o conjunto de treinamento e S o estoque de dados, o lado direito da Equação 4.4 é dominado por $\delta_{T \to S}$ se o estoque de exemplos for grande. A função objetivo proposta (K-Covers), baseada nas premissas apresentadas, é então a minimização do raio de cada cluster.

A solução K-Covers apresenta uma limitação para o tamanho do estoque de dados, que deve ser grande o bastante para manter as condições teóricas estabelecidas. Adicionalmente, o algoritmo de seleção adquire apenas uma instância por cluster, implicando que os dados devem ser divididos em tantos clusters quanto o número de elementos desejados. Em patologia, essa abordagem pode não fornecer ganhos de aprendizado dado que os tipos de tecido ou conjunto de características marcantes podem não ser suficientes pra se produzir números elevados de clusters (ver Figura 7.21).

O trabalho desenvolvido por *Yuan et al.* [96] também possui relevância para o modelo desenvolvido no presente trabalho. Os autores argumentam que o processo de aprendizado ativo deve ser usado a partir de uma combinação de estratégias de aquisição, como ponderações por incertezas, abordagens geométricas e de similaridade, combinadas de maneira dinâmica. Os autores argumentam que o desbalanceamento do desempenho de classificação observado sobre as diferentes classes presentes em um banco de dados possui efeitos negativos para o aprendizado ativo e que o uso de uma estratégia de seleção única deixa de ser efetiva à medida que as iterações de aquisição vão sendo executadas.

Nesse trabalho, *Yuan et al.* utilizam dois grupos de métricas para avaliar a capacidade informativa dos exemplos: métricas relativas ao conjunto de exemplos rotulados e métricas relativas ao modelo atual. O grupo que considera o conjunto de exemplos rotulados emprega medidas de densidade e similaridade, enquanto o grupo relativo ao modelo utiliza incertezas e métricas relativas ao desempenho por rótulos (referir a [96] para mais detalhes). Esses grupos são balanceados na seleção de exemplos através de um parâmetro α, que considera que o grau informativo entre ambos os grupos é independente. Os autores denominaram sua solução como MCDAL.

A proposta foi testada sobre as bases de dados MNIST e CIFAR-10 e apresenta ganhos sobre um conjunto de outras propostas avaliadas, entre elas: MC Dropout utilizando BALD como métrica de incertezas, seleção pela entropia máxima e seleção aleatória. Dentre essas, se destaca como a mais relevante a implementação de MC BALD, sendo superada pelo MCDAL, todavia com uma margem pequena. Apesar dos ganhos apresentados, o tempo de execução do método constitui uma limitação. Os autores definem que o algoritmo de seleção apresentado possui tempo de execução $\mathcal{O}(N|D_U|)$, onde N é o número de exemplos adquiridos e D_U é o conjunto de exemplos não rotulados. Apesar de não ter sido apresentada uma avaliação temporal mais precisa, a passagem sobre o conjunto não

rotulado múltiplas vezes, com sucessivos cálculos de métricas informativas tende a tornar a solução pouco aplicável a casos em que N e/ou D_U sejam grandes.

4.2 Simplificação de CNNs

As CNNs são classificadores extremamente versáteis e efetivas (ver Seção 2.2.1), todavia possuem altas exigências computacionais, dependendo, na prática, de processadores específicos ou Graphics Processing Units (GPUs) para serem aplicadas [127]. A simplificação de CNNs é uma área de pesquisa muito ativa, contando com ampla variedade de soluções disponíveis na literatura [128, 129]. Essas soluções são voltadas para a redução dos custos computacionais exigidos por modelos de *deep learning*, com algumas delas focando, inclusive, em permitir seu uso em dispositivos móveis [130, 131].

A aplicação de técnicas de simplificação de CNNs junto ao DADA pode trazer ganhos no tempo de cada iteração do aprendizado ativo. Neste trabalho, foi desenvolvida uma solução específica para ser usada com sistemas de AL (ver Seção 6.3), de forma que é importante abordar os fundamentos das principais técnicas de simplificação de redes disponíveis na literatura, bem como as razões que levaram ao desenvolvimento dessa solução específica.

Os métodos de simplificação podem ser classificados como não estruturais ou estruturais [131]. As abordagens não estruturais não removem nenhum dos componentes das redes (canais, camadas ou conjuntos de camadas). Essas soluções atuam cortando conexões entre os neurônios de camadas adjacentes, através da anulação de certos pesos, produzindo um sinal de saída igual a zero para essas conexões. Esse tipo de abordagem era mais utilizado sobre modelos relativamente pequenos, sendo pouco efetivo sobre os modelos atuais, especialmente em relação ao consumo de memória, que permanece inalterado com a mera anulação de pesos. Adicionalmente, o esparsamento de pesos, com a remoção de valores individuais nos filtros pode até prejudicar o desempenho da rede, visto que as bibliotecas e o hardware disponíveis são projetados para trabalhar com estruturas de dados de tamanhos que se mantém dentro de padrões (matrizes $N \times N$, por exemplo).

As soluções de simplificação estruturais excluem e/ou reduzem a quantidade de componentes das redes, sendo a remoção de canais a estratégia mais utilizada na literatura [129, 132, 133, 134]. Essa estratégia atua sobre a dimensão da largura de cada camada convolucional, sendo que a seleção de quais e quantos canais serão removidos se dá de diferentes maneiras, conforme a solução desenvolvida. Um ponto em comum entre as abordagens discutidas a seguir é que o processo de simplificação deve partir de um modelo pré-treinado ou deve ser realizado durante o treinamento de uma rede, criando

uma versão reduzida que é específica à aplicação e ao conjunto de dados de treinamento utilizado.

Zou et al. [134] utilizam uma rede pré-treinada semelhante à VGG e produzem um índice discriminador para cada mapa de características gerado após a operação de convolução. Esse índice representa um fator de contribuição de cada mapa para o desempenho de classificação da rede. Além de desenvolverem esse índice, os autores também definem pontos críticos para cada camada convolucional, os quais indicam a quantidade de mapas que podem ser removidos, considerando os pontos de inflexão dos gradientes das curvas de discriminação. Com base nesses dois valores, é possível remover quantidades específicas de mapas de características de cada camada (bem como dos filtros das camadas subsequentes), com níveis de impacto de classificação correspondente ao ponto crítico de corte. Entre os resultados obtidos, é interessante destacar que as camadas mais baixas da rede apresentaram os menores índices de poda, podendo perder até 15,6% de seus mapas, enquanto as camadas mais altas poderiam sofrer podas mais intensas, de até 59,7% de seus mapas.

Já Luo et al. [135] focam na remoção de filtros, tendo como consequência a remoção dos canais de saída de uma camada convolucional. Um dos diferenciais do método desenvolvido pelos autores, denominado *ThiNet*, é que a escolha dos filtros a serem removidos advém de estatísticas realizadas sobre a saída da camada subsequente ao filtro que se avalia remover. A seleção dos filtros a serem removidos advém da identificação de um subconjunto $S \subset \{1, 2, ..., C\}$ que mantém as condições da equação 4.5:

$$\begin{aligned} \hat{y} &= \sum_{c \in S} \hat{x}_c; \\ \hat{x}_c &= \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \widehat{\mathcal{W}}_{c,k_1,k_2} \times x_{c,k_1,k_2}, \end{aligned} \tag{4.5}$$

dada uma camada i dotada de C canais de entrada, cujos pesos $\widehat{\mathcal{W}} \in \mathcal{R}^{C \times K \times K}$ se distribuem sobre o mesmo número de canais, com campo receptivo de $K \times K$. Ainda, $\hat{x}_c$ corresponde ao resultado da operação de convolução sobre a camada c do tensor de entrada $x \in R^{C \times K \times K}$ e $\hat{y}$ é o resultado da convolução sobre todo esse tensor. Considerando um conjunto de m exemplos de entrada para o sistema (que podem ser diferentes imagens), procura-se encontrar o mínimo conjunto S que continue mantendo o valor $\hat{y}$. Os canais $c \notin S$ são considerados desnecessários e podem ser removidos.

A solução *ThiNet*, assim como as demais, depende de um modelo pré-treinado para gerar os valores $\hat{y}$. Adicionalmente, é necessário realizar m previsões para a obtenção do conjunto mínimo S e também uma fase de pós-processamento para ajuste fino dos filtros remanescentes, durante a qual uma versão especializada de convolução em grupo é usada.

Osaku et al. [136] propuseram duas soluções com treinamento progressivo do modelo, denominados *objective Pruning with Progressive Retraining (oPPR)* e *subjective Pruning with Progressive Retraining* (sPPR). O principal diferencial dessas abordagens, com relação ao custo de simplificação, é se evitar repetidos treinamentos do modelo original e/ou do modelo simplificado por completo, executando o treinamento apenas das camadas 1 a $l + 1$ para cada camada simplificada l. Apesar de procurar diminuir o custo de retreinamento dos modelos reduzidos, as soluções ainda demandam l seções de treinos, sobre trechos progressivos do modelo. Destaca-se também que a versão subjetiva (sPPR) da solução permite que um projetista de CNNs visualize os mapas de ativação médios gerados sobre instâncias da mesma classe, permitindo que as regiões de ativação que os filtros produzem sobre as imagens sejam analisadas visualmente e, assim, filtros que não conseguem distinguir regiões diferentes de classes diferentes são considerados supérfluos.

As soluções discutidas até aqui focam especificamente nos filtros convolucionais e/ou nos mapas de características gerados por eles, removendo ambos, a partir de modelos de decisão distintos. Um dos motivos para que as soluções de simplificação foquem exclusivamente em filtros/mapas de características é que a remoção de estruturas maiores, como canais ou blocos, na verdade geram uma nova rede, ao invés de simplificar uma rede original [135]. Em tese, há uma linha tênue que separa as técnicas de simplificação estruturais que abordam componentes além de filtros das técnicas de *Network Architecture Search (NAS)* [137, 138] ou AutoML [139] das técnicas de simplificação mais agressivas. A característica mais marcante dessas técnicas de simplificação, e que as difere de NAS ou AutoML, é o escopo de "busca" em que a simplificação atua. Nesse último caso, o ponto de partida é sempre único: uma rede complexa em específico, a partir da qual se busca produzir um modelo mais barato.

Definida essa diferenciação, dos trabalhos mais próximos da solução de simplificação de CNNs aqui proposta (Seção 6.3.2) está o *Generative Adversarial Learning (GAL)* [140]. O GAL é uma solução que remove estruturas heterogêneas redundantes, como filtros, blocos e ramos inteiros de CNNs. Outro fator importante é que o GAL não demanda retreinamenos iterativos das redes, executando o processo fim a fim em uma sequência única, demandando, todavia, um modelo inicial pré-treinado. A redução se dá a partir da aplicação de uma máscara sobre a arquitetura original, a qual define quais estruturas serão removidas. Essa máscara é aprendida durante uma sequência de treinamento em que a saída da rede reduzida é otimizada para se assemelhar à saída da rede original. Apesar de ser uma abordagem promissora, sua implementação ainda é complexa e custosa para o uso no contexto do aprendizado ativo.

Outra solução, que foi capaz de produzir uma rede reduzida sem perdas de desempenho de classificação em relação ao modelo original, é apresentada por Ding et al. [141]. A

relevância dessa solução para o aprendizado ativo é grande, visto que um dos objetivos na produção de modelos reduzidos é que eles obtivessem o desempenho mais próximo possível da CNN original. A importância desse resultado é melhor discutida na Seção 6.3.2, todavia, em breve resumo, considera-se a hipótese de que a equivalência de resultados entre modelos seja um indicador de que incertezas produzidas por um deles possam ser aproveitadas pelo outro. Adicionalmente, em seu trabalho, os autores avaliaram sua solução contra diversas outras disponíveis na literatura. Dessa forma, essa solução é estudada de maneira mais aprofundada a seguir.

ResRep [141] é uma estratégia de poda de CNNs que se encontra no estado da arte, utilizando reparametrização de pesos para reduzir a largura de uma rede. Ela implementa uma solução em duas etapas, referidas como *lembrança* e *esquecimento*, inspiradas por pesquisas em neurobiologia. Essas pesquisas, conforme os autores, indicam que os tecidos neuronais vivos são regidos por mecanismos independentes para aprender novas conexões e posteriormente as esquecer, em um processo de otimização de eficiência energética e espacial. Dessa forma, Ding et al. produziram uma solução que busca mimetizar essa lógica.

Na etapa de lembrança (Rep), é necessário partir de um modelo original pré-treinado, que posteriormente será retreinado com a adição de camadas de *compactadores*, associadas às camadas convolucionais originais. O objetivo é identificar filtros que pouco contribuem com o processo de aprendizagem. Os compactadores são camadas convolucionais de 1×1 que aplicam penalidades aos gradientes ou "redefinição de gradientes", fazendo com que seus valores, em algumas camadas, se aproximem de zero.

Durante a redefinição de gradientes, uma penalidade tipo grupo Lasso é usada em conjunto com a função objetivo do treinamento para produzir pesos esparsos abrangendo todos os parâmetros de determinados filtros. A operação de redefinição dos gradientes é formulada pela Equação 4.6.

$$\begin{aligned} L_{total}(X,Y,\Theta) &= L_{perf}(X,Y,\Theta) + \lambda P(K); \\ G(\mathbf{F}) &= \frac{\partial L_{total}(X,Y,\Theta)}{\partial \mathbf{F}} \leftarrow \frac{\partial L_{perf}(X,Y,\Theta)}{\partial \mathbf{F}} * m + \lambda \frac{\mathbf{F}}{||\mathbf{F}||_E}; \\ ||\mathbf{F}||_E &= \sqrt{\sum_{c=1}^{C}\sum_{p=1}^{K}\sum_{q=1}^{K} F_{c,p,q}^2}. \end{aligned} \tag{4.6}$$

Na equação, L_{total} é a função objetivo aplicada à entrada X com rótulo Y, dados os pesos atuais da rede, Θ. O λ é um fator de intensidade da penalidade e $P(K)$ é a penalidade Lasso adicionada à função de custo regular, L_{perf}. Os gradientes correspondentes a L_{perf}, para cada filtro ($\mathbf{F}$) da camada convolucional, podem ser anulados com a aplicação de

uma máscara binária m. O uso dessa máscara evita que os termos correspondentes à função de custo e ao fator de penalidade entrem em uma competição entre si durante o retreino, facilitando que os gradientes e, consequentemente os pesos de determinados filtros, tendam a zero. Ao mesmo tempo, a definição da máscara para cada filtro de cada compactador irá especificar quais serão os filtros da camada convolucional original que serão removidos.

A definição da máscara se dá durante o processo de retreinamento do modelo, após já aplicados os compactadores. Essa definição se dá com base no conceito de "avaliação de importância baseada em competência" (*competence-based importance evaluation*), que busca identificar qual termo do gradiente $G(\mathbf{F})$ se sobressai sobre o outro, ou seja, se a penalidade sobre os pesos terá mais relevância do que a otimização da função de custo $L_{perf}(X, Y, \Theta)$. Assim, a importância de cada filtro de cada camada compactadora é armazenada em uma matriz $\mathcal{M}$, conforme a equação 4.7:

$$\mathcal{M} \ = \ \{(i,j) \rightarrow ||Q^{(i)}_{j,:}||_2 | \forall \ 1 \leq i \leq n, 1 \leq j \leq D^{(i)}\}, \tag{4.7}$$

considerando a existência de n camadas compactadoras, cujos filtros são $Q^{(i)} \in R^{D \times D}$.

As máscaras $m^{(i)}_j$ (do filtro j da camada i) são definidas como zero, de maneira iterativa, até que um limite de θ filtros sejam escolhidos para remoção da camada i ou até que um objetivo de redução em termos de custo computacional seja atingido. O gradiente final $G(\mathbf{F})$ (ver equação 4.6) é então comparado a um valor limite (ϵ). Caso o valor do gradiente seja abaixo desse limite, o filtro é removido. Espera-se que $G(\mathbf{F})$ seja próximo de zero para filtros cuja máscara binária é zero, já que nesse caso apenas penalidades são aplicadas aos gradientes. Após a identificação dos canais a serem removidos, a etapa de esquecimento é concluída com a reconstrução do modelo original, mas removendo alguns dos canais de certas camadas.

Ambos componentes (lembrança e esquecimento) são necessários para se manter a resistência do modelo, definida como a capacidade de manter alta performance de classificação frente à mudança do objetivo de otimização durante o treinamento, e para se manter a capacidade de poda, que é a manutenção dos níveis de desempenho frente à redução da rede.

Em seu artigo, Ding et al. realizaram experimentos aplicando o ResRep sobre a rede ResNet50. Adicionalmente, os autores compararam seus resultados com um grande número de outras soluções de simplificação, inclusive aquelas apresentadas em [135, 140]. Como demonstração do comportamento do ResRep, foi formulada a Tabela 4.1, correspondente à rede ResNet50 V2 [142] e contendo as arquiteturas das redes reduzidas avaliadas no presente trabalho (ver Seção 7.2.6), obtidas a partir da utilização de diversos níveis de redução (dados pelo parâmetro ϵ). Nessa tabela, o símbolo $\mathbb{P}$ marca as camadas em que

foram executadas podas de canais.

CNN	$\epsilon = 0.82$	$\epsilon = 0.84$	$\epsilon = 0.86$	$\epsilon = 0.88$	$\epsilon = 0.90$	$\epsilon = 0.92$	$\epsilon = 0.94$
ResNet50 V2 [142] # Params	12,527,836	9,421,008	8,663,740	9,225,475	7,931,287	4,882,052	4,696,612
Conv 1	7 x 7, 64, stride 2						
Estágio 1	[1x1, 64 3x3, 64 1x1, 256] x3	[1x1, 64 3x3, 64 1x1, 256] x3	[1x1, 64 3x3, 64 1x1, 256] x3	[1x1, 64 3x3, 64 1x1, 256] x3	[1x1, 64 3x3, 64 1x1, 256] x3	[1x1, 64 3x3, 64 1x1, 256] x3	[1x1, 64 3x3, 64 1x1, 256] x3
Estágio 2	[1x1, 128 3x3, 128 1x1, 512] x4	[1x1, 128 3x3, 128 1x1, 512] x4	[1x1, 128 3x3, 128 1x1, 512] x4	[1x1, 128 3x3, 128 1x1, 512] x4	[1x1, 128 3x3, $\mathbb{P}$ 1x1, 512] x4	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 512] x4	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 512] x4
Estágio 3	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 1024] x6	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 1024] x6	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 1024] x6	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 1024] x6	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 1024] x6	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 1024] x6	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 1024] x6
Estágio 4	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 2048] x3	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 2048] x3	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 2048] x3	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 2048] x3	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 2048] x3	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 2048] x3	[1x1, $\mathbb{P}$ 3x3, $\mathbb{P}$ 1x1, 2048] x3

Tabela 4.1: Número de parâmetros e camadas para as redes reduzidas pelo ResRep (classificação binária). Em cada nível de redução, $\mathbb{P}$ indica as posições dos blocos em que houve poda de canais.

Da tabela, é importante destacar que o ResRep prioritariamente selecionou filtros a serem removidos das camadas finais da rede, de maneira semelhante à solução em [134]. No Estágio 1, nenhuma camada sofreu redução do número de canais. É interessante destacar também que a seleção de quais camadas serão removidas depende tanto de um pré-treinamento da rede original quanto do treinamento executado para a execução da redefinição de gradientes. Dessa forma, dadas diferentes inicializações de pesos e diferentes conjuntos de imagens de entrada, as redes reduzidas obtidas podem variar entre múltiplas execuções e o número de parâmetros do modelo para cada valor de ϵ pode inclusive oscilar positivamente, como ocorre entre $\epsilon = 0.86$ e $\epsilon = 0.88$.

O ResRep é uma solução robusta, formulada sobre um sólido arcabouço teórico e, conforme seus autores, foi capaz de produzir os mesmos níveis de desempenho de classificação da rede original com uma redução de 54,54% do total de *Floating Point Operations (FLOPs)*. Assim, é uma referência importante a se comparar com a solução desenvolvida no presente trabalho, mesmo que os objetivos de ambas não seja o mesmo.

Capítulo 5

Aplicação Motivadora

O presente trabalho é motivado principalmente pela análise de imagens de *slides* digitalizados de tecidos de biópsias (WSIs), mais especificamente por aplicações para detectar e classificar padrões espaciais de estruturas e objetos em tecidos de biópsias. Nesse sentido, são utilizadas aplicações baseadas em *deep learning*, originalmente desenvolvidas em [143, 144] para identificar e classificar padrões de LITs, como base para os testes e avaliações da abordagem de aprendizado ativo.

Nessa seção são apresentados os conceitos fundamentais envolvidos na aplicação foco, bem como as soluções que originalmente produziram resultados de classificação sobre as imagens de biópsias ora utilizadas e o processo de criação do banco de dados que alimenta as soluções testadas.

5.1 Linfócitos Infiltrantes de Tumor (LITs)

Os linfócitos são um grupo de células brancas do sangue, sendo parte do sistema imunológico humano. Existem diversos subtipos de linfócitos, com características e atuações diferentes sobre áreas de tumor, bem como modos de atuação distintos sobre diferentes tipos de câncer [145]. As interações entre os tipos de células e os tipos de tumor são complexas e passam por intenso estudo, com foco tanto no desenvolvimento de tratamentos quanto na acurácia de diagnósticos e prognósticos.

Diversos estudos demonstraram que existem correlações entre padrões de LITs e a sobrevida de pacientes de múltiplos tipos de câncer, com casos em que a alta densidade de LITs se relaciona com melhores resultados clínicos [146, 147]. Adicionalmente, a localização das concentrações de LITs e as combinações entre a presença ou ausência de subtipos de LITs também são variáveis que influenciam na resposta a tratamentos baseados em imunoterapia [148, 149].

Dessa forma, uma análise quantitativa da organização espacial dos linfócitos, com respeito ao micro-ambiente do tumor, pode fornecer importantes evidências da interação entre o câncer e o sistema imunológico, além de valiosos biomarcadores para melhor prever a resposta a tratamentos e o prognóstico geral.

Imagens de tecidos de biópsias capturados por *scanners* de alta resolução contém detalhes e informações espaciais de grande riqueza sobre a morfologia do tecido no nível subcelular. Assim, tem havido um crescente interesse em métodos de análise de imagens desses tecidos, com diferentes focos e objetivos. Uma visão dos trabalhos realizados na área pode ser encontrada em vários artigos de alta qualidade [150, 151, 152, 153, 154, 155].

A capacidade de análise das imagens geradas pela digitalização de tecidos é, portanto, fundamental para que avanços práticos sejam obtidos com relação aos estudos relacionados a LITs. Como já exposto no Capítulo 1, os slides digitalizados - WSIs -, usualmente são imagens que contém bilhões de pixels, capturando da ordem de milhares de microestruturas e milhões de objetos subcelulares, como núcleos de células. Soma-se a isso o fato de que de um mesmo paciente normalmente são extraídos múltiplas amostras de tecidos, que darão origem a um conjunto de imagens.

O exame manual de quantidades de informação dessa magnitude é inviável de um aspecto de custo de mão de obra e suscetível a erros, o que gera taxas de discordância intra-especialista e entre especialistas, algo prejudicial na produção de diagnósticos. Assim, para que seja possível obter ganhos para as avaliações médicas já existentes e, também, para que seja possível o desenvolvimento de novas metodologias e inovações clínicas, é fundamental que as imagens geradas possam ser analisadas por completo e de maneira automática.

5.2 Fluxo de Trabalho da Aplicação

Métodos recentes de análises de imagens de patologia empregam modelos de *deep learning* para identificar características e padrões espaciais de objetos e micro-estruturas nessas imagens [143, 144, 156, 157, 158, 159, 160]. Os trabalhos realizados por *Saltz et al.* [143] e *Shahira et al.* [144] propõem um fluxo de trabalho baseado em CNNs para a análise de padrões de LITs em imagens de tecidos manchados com hematoxilina e eosina (H&E). Esse fluxo de trabalho permite que a análise de LITs inclua a quantificação de células no tecido bem como a identificação de padrões espaciais complexos para avaliar a interação entre LITs e regiões cancerígenas.

O fluxo de trabalho de análise desenvolvido por *Saltz et al.* [143] é dividido em duas fases, conforme apresentado na Figura 5.1, e serve de base para a metodologia aqui proposta e descrita a seguir. A fase inicial consiste do treinamento do modelo CNN, conforme

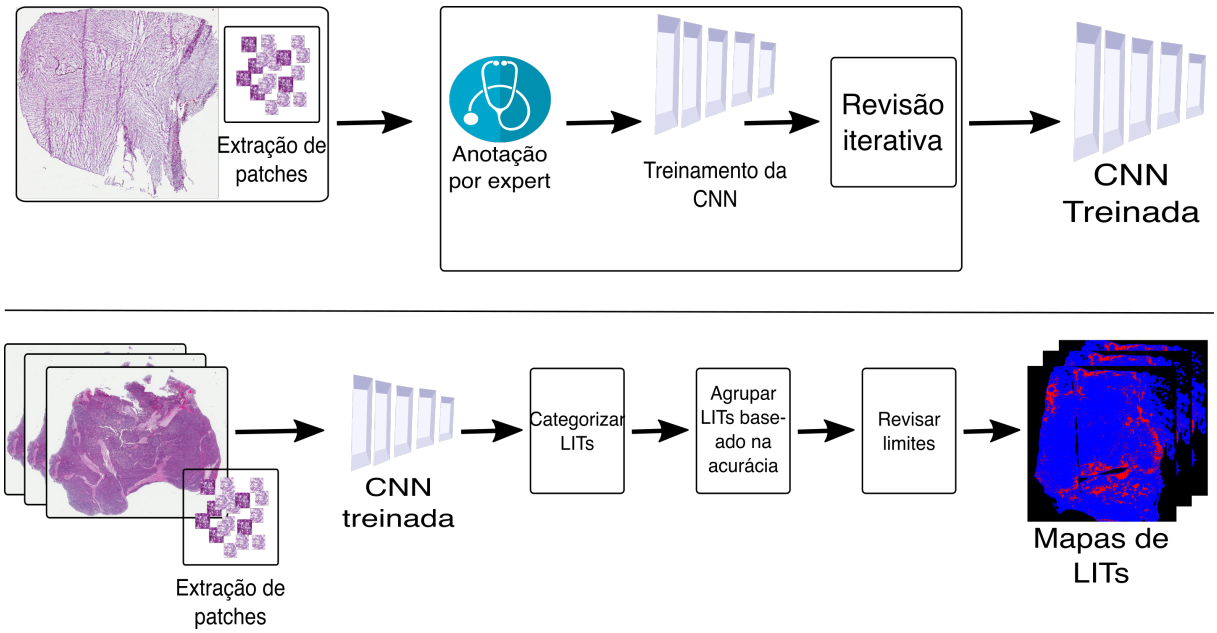


Figura 5.1: Fluxo de trabalho desenvolvido por *Saltz et al.*.

mostrado na porção superior da figura, com o objetivo de identificar porções das imagens consideradas LIT positivos/negativos. Essa fase produzirá um conjunto inicial de treinamento bem como um modelo já treinado, após etapas sucessivas de anotações, em que o *expert* tem liberdade para escolher regiões para anotação. Nessa fase há o treinamento regular da rede, sem a aplicação de técnicas de aprendizado ativo. Os *patches* são extraídos de múltiplos slides após a identificação das regiões de interesse e são fornecidos a outros patologistas para que os anotem e possam integrar o conjunto de treinamento para o modelo de classificação. Os *patches* formados dessa pré-seleção correspondem a secções de 50 x 50 micrômetros do tecido (*patches*), extraídos de WSIs.

Em uma segunda fase, esse modelo, após treinado, é aplicado aos *patches* de entrada (porção inferior da figura) de outros slides para que sejam criados os mapas de LITs. Assim, o conjunto de treinamento inicial pode ser expandido através de uma etapa de classificação automática, que passará por uma revisão dos limites das regiões positivas por um *expert*.

Para a aplicação desse fluxo de trabalho na referida solução, uma abordagem colaborativa em pequena escala foi utilizada, em conjunto com um processo de refinamento iterativo, para que fosse possível obter *patches* anotados. Vários patologistas se voluntariaram para gerar um conjunto de treinamento inicial, revisar imagens de slides de tecidos e fornecer a classificação dos *patches* como LIT-positivos ou negativos. A partir desse conjunto inicial foi possível treinar o primeiro modelo, que foi posteriormente aplicado sobre slides ainda não utilizados. Em seguida foi necessário que os patologistas revisassem os resultados gerando novos mapas de LITs bem como anotassem novos conjuntos de *patches*

para incrementar o conjunto de treinamento.

De uma perspectiva dos métodos de análise, o trabalho desenvolvido em [143] mostrou que (1) um conjunto de treinamento inicial, mesmo que gerado por múltiplos patologistas, não necessariamente resulta em modelos robustos e precisos o suficiente; e (2) o refinamento iterativo, com aumento do conjunto de treinamento, ajuda a melhorar o desempenho dos algoritmos para esse tipo de análise. O processo iterativo, todavia, depende de múltiplos patologistas para anotarem imagens em múltiplos passos. O DADA busca fazer com que esse tipo de processo seja mais sistemático, reduzindo a carga de trabalho do revisor/anotador e acelerando o processo como um todo através da seleção inteligente de conjuntos de *patches* que vão otimizar o modelo de maneira mais significativa.

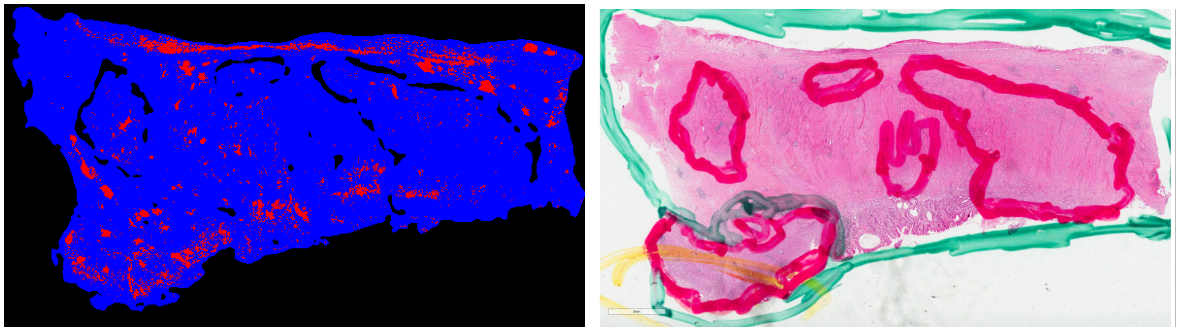
O trabalho desenvolvido em [143] propiciou um ponto de partida para a criação de bases de dados com o tamanho necessário para que se atinja, de maneira efetiva, o objetivo de ter modelos com elevada precisão de classificação na aplicação em estudo. As anotações automáticas produzidas com essa solução sobre grande número de WSIs (5.202 imagens), e a geração de mapas de probabilidades (*heatmaps*) com a indicação das regiões com maiores concentrações de LITs, fez com que fosse possível conduzir os experimentos do DADA, sendo gerados *patches* anotados conforme descrito a seguir.

5.3 Geração da Base de Dados de LITs

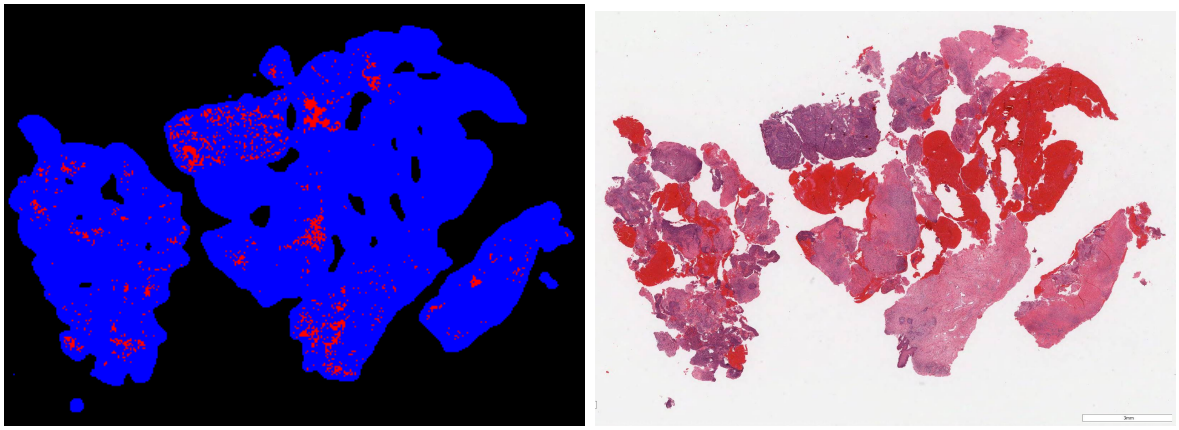
O modelo produzido a partir do trabalho realizado por [143] foi aplicado a 13 tipos de câncer, para um total de 5.202 slides H&E, em que diversas associações de LITs, entre características moleculares, sobrevida do paciente e características do câncer puderam ser feitas com sucesso. Os mapas de probabilidade gerados permitiram também que, no presente trabalho, fosse criada uma base de dados com as classificações de *patches* de um conjunto de slides, de maneira a formar uma extensa base rotulada, que pudesse servir para a simulação de um ambiente de aprendizado ativo.

A base de dados utilizada nos experimentos conduzidos é composta de 62 slides entre os 5.202 disponíveis como resultado do trabalho citado. Os mapas de probabilidades gerados para esses 62 slides permitem que cada *patch* produzido a partir desses slides receba uma classificação positiva/negativa quanto à presença de LITs. A Figura 5.2 ilustra alguns dos mapas produzidos e as imagens originais correspondentes, de forma exemplificativa.

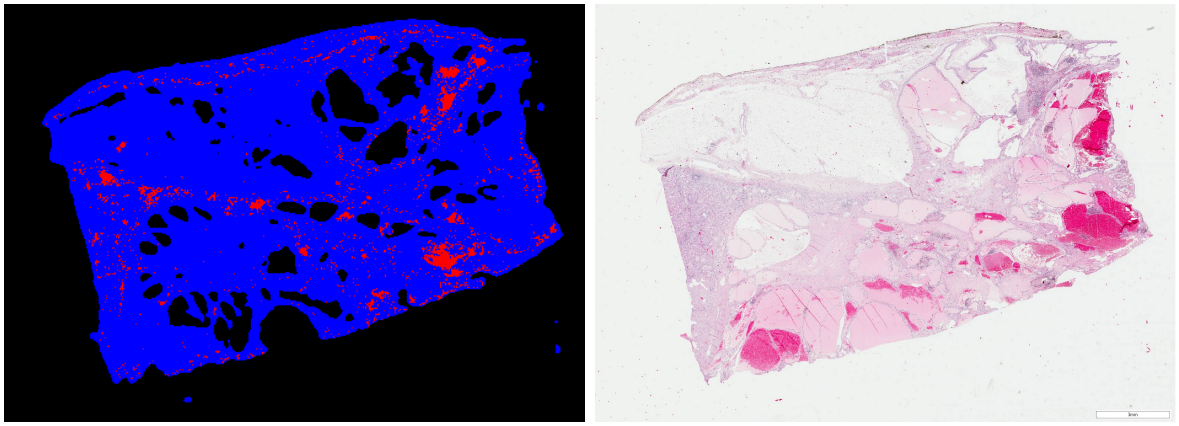
Os mapas de probabilidade produzidos são versões em tamanho reduzido dos slides originais, de forma que, para a correta classificação dos *patches*, é necessário identificar a região do mapa correspondente a cada *patch* extraído do WSI. A partir da coloração que a região correspondente ao *patch* possui no mapa é possível definir a classificação dos *patches*.



(a) TCGA-BR-8058-01Z-00-DX1.



(b) TCGA-C5-A7CK-01Z-00-DX1.



(c) TCGA-S5-AA26-01Z-00-DX1.

Figura 5.2: Mapas de LIT para três slides (porção esquerda das figuras) e o slide original (porção direita das figuras). As regiões em vermelho demarcam localizações positivas. Em azul porções de tecido negativas. A área preta corresponde a fundo de imagem.

A correspondência entre a região em que os *patches* foram extraídos e os mapas de probabilidade dependem do grau de magnificação em que foi feita a captura dos slides. Considerando as configurações utilizadas para produzir os *heatmaps* em [143], cada pixel dos mapas corresponde a uma área de 50×50 μm de tecido, o que, em uma magnificação de 20×, corresponderia a 0,5 micrômetros de tecido por pixel e geraria *patches* de 100×100

pixels.

No presente trabalho, optou-se por utilizar *patches* que cobrissem uma área maior de tecido, a fim de permitir que as redes capturassem mais informações por imagem, o que contribuiria para o aprendizado, especialmente em cenários em que o conjunto de treinamento fosse pequeno. Assim, optou-se por extrair dos slides *patches* correspondentes a 150×150 μm. Nessa nova configuração, cada *patch* é mapeado a uma região de 3×3 pixels no *heatmap* correspondente.

Assim, a classificação LIT-positivo/-negativo de cada *patch* depende da definição de uma proporção entre pixels positivos e negativos contidos na região considerada. A concentração de linfócitos presentes em regiões de tecidos usualmente é classificada de maneira qualitativa em 4 grupos, com denominações variáveis que podem ser compiladas nas seguintes categorias: i) escasso ou ausente; ii) moderado; iii) denso; iv) muito denso [161, 162]. Dessa forma, em se considerando uma distribuição linear das densidades, foi adotado a presença de ao menos 20% de regiões positivas em relação às negativas para classificar o *patch* como positivo.

Os *patches* extraídos foram redimensionados para ocupar uma região de 300 × 300 pixels. Essa base de dados é altamente desbalanceada, apresentando um total de 525.431 *patches*, dos quais 16% são positivos para a presença de LITs. Apesar da base de dados possuir um total expressivo de *patches*, sendo suficiente para se treinar adequadamente modelos com diferentes propósitos, a produção de rótulos passa por um processo de classificação automático das regiões, produzida pelo modelo descrito na seção anterior. Essa característica, aliado ao erro inerente ao processo aqui adotado para considerar um *patch* como positivo ou negativo, adiciona ruído ao banco de dados, que certamente impacta no desempenho de qualquer modelo treinado a partir dele. Esses efeitos adversos podem ser, em parte, amenizados pelo uso de técnicas como *data augmentation*, que atuam melhorando a capacidade de generalização dos modelos reduzindo a ocorrência de *overfitting*.

A aplicação de *data augmentation* no contexto da histopatologia e especificamente em problemas de identificação de LITs já foi realizada em trabalhos anteriores [163] e fornecem informações importantes para sua aplicação junto ao DADA. No trabalho citado, *Le et al.* aplicaram *data augmentation* sobre um conjunto de treinamento composto de 86.154 *patches* positivos/negativos quanto à presença de LITs. As operações efetuadas foram: de normalização ao redor de uma média de 0,0 e desvio padrão 1,0; rotação aleatória entre 0 e 22,5 graus; espelhamento horizontal e vertical; e alterações de brilho, contraste e saturação. Essas mesmas operações foram utilizadas em experimentos executados com o DADA, sendo discutidas e demonstradas no Capítulo 7.

Considerando o exposto, o esforço em se ter um banco de dados totalmente anotado por especialistas, produzido de maneira ótima, com o mínimo de ruído, e de tamanho

suficiente para permitir o treinamento de modelos com o desempenho desejado é a grande motivação para o desenvolvimento do trabalho aqui realizado.

Capítulo 6

Diversity-Aware Data Acquisition (DADA)

Neste capítulo é descrita a solução de aprendizado ativo elaborada, tendo como foco aplicações de processamento de imagens patológicas utilizando *deep learning*. Essa solução busca tanto a minimização do número de imagens presentes no conjunto de treinamento, ainda produzindo alto desempenho de classificação, quanto a redução do tempo de execução demandado para se gerar o conjunto de *patches* que deverão ser rotulados e utilizados no treinamento para a iteração seguinte do AL.

O presente capítulo descreve a solução desenvolvida dentro do escopo deste trabalho bem como uma solução gráfica para a utilização do DADA em ambientes reais (Seção 6.1). Em seguida, ponderações acerca do tempo de execução obtido por essa solução são explicitados na Seção 6.2 e as melhorias elaboradas para a diminuição desse tempo na Seção 6.3.

6.1 Diversity-Aware Data Acquisition (DADA)

Em trabalhos anteriores [23, 24], ficou demonstrado que a aplicação direta de métodos já consagrados de AL ao contexto da patologia digital apresenta resultados apenas ligeiramente superiores à seleção aleatória. Durante os trabalhos aqui conduzidos, foi observado que um dos aspectos que afetam o desempenho dos métodos de aprendizado ativo baseados unicamente em incertezas é a seleção de exemplos que apresentam propriedades morfológicas semelhantes, reduzindo sua efetividade para o treinamento.

Foi observado que as funções de incertezas apresentadas na Seção 4.1 podem selecionar conjuntos de *patches* com pouca diversidade, considerando a aplicação motivadora apresentada. A cada etapa de aquisição, caso apenas os *patches* de maior incerteza sejam selecionados, estes tendem a ser similares em termos de características de tecidos. Essa seleção, aquém do ótimo, pode prolongar o processo de aprendizado ativo, dado que o

conjunto de treinamento é aumentado com novos *patches* que provavelmente detém uma distribuição informativa similar àqueles que já se encontram no conjunto.

Motivado por essas observações, uma nova função de aquisição é proposta, denominada DADA. Para garantir que *patches* com diferentes aspectos sejam selecionados em uma etapa de aquisição, DADA os agrupa em conjuntos (*clusters*) baseado em suas *features* ou características e então executa a seleção em cada conjunto formado. Ao se escolher os *patches* com maior incerteza de cada *cluster*, DADA permite a adição de imagens com características diversas e que ainda são desafiadoras para a rede CNN classificar.

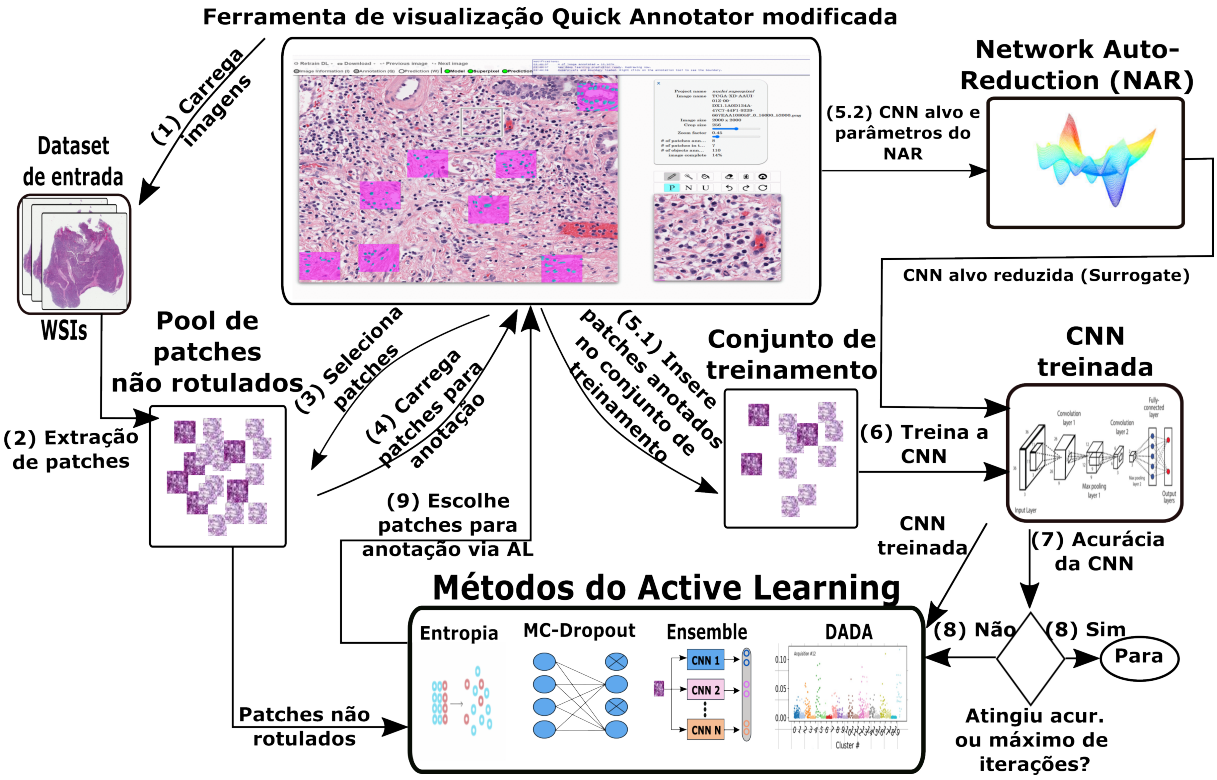


Figura 6.1: Fluxo de trabalho aplicado pelo DADA.

O processo executado pelo DADA em cada aquisição é apresentado no Algoritmo 2 e na Figura 6.1 (NAR e a ferramenta gráfica serão abordados a seguir). Ele é organizado em três passos principais: cálculo de uma métrica de incertezas, cálculo de uma métrica de densidade de informação e execução de uma lógica de combinação para determinar a seleção final. Deve-se notar que o DADA não propõe uma nova medida de incertezas, sendo usadas aquelas já disponíveis na literatura, tanto com a abordagem por Ensembles quanto a Bayesiana para a formulação dessas métricas, como já apresentadas na Seção 4.1.

Em conjunto com o cálculo de incertezas, o DADA extrai características a partir de camadas profundas da CNN, para cada *patch* presente no *pool*. Essas características são usadas por serem conhecidamente mais efetivas em tarefas de visão computacional, podendo ser facilmente obtidas com uma única passagem sobre os dados e extraídas da

última camada convolucional das redes. A etapa de extração de *features* corresponde à linha 1 do Algoritmo 2. O *pool* de *patches* (U) é recebido como entrada. Um vetor de incertezas (PU) e as caraterísticas extraídas (DF) são computados abarcando todos os *patches*.

A medida de densidade de informação usada no DADA é implementada nas linhas 2 e 3 do Algoritmo 2. Os vetores com as características (DF) extraídos de CNNs modernas (como a InceptionV4), usualmente formam estruturas de dados de alta dimensionalidade. A Análise de Componente Principal (*Principal Component Analysis (PCA)*) é executada para reduzir as dimensões do vetor de características e melhorar o desempenho do agrupamento das imagens (linha 2). A redução de dimensionalidade pode ser executada diretamente pelas redes, através da inserção de camadas de *pooling* ou pelo ajuste dos valores de *stride* das camadas convolucionais. Todavia, a inserção dessas alterações nas redes afeta também o processo de treinamento, podendo gerar impactos no desempenho de classificação, motivo pelo qual se optou por manter as arquiteturas das redes usadas e aplicar o PCA em etapa posterior.

Algoritmo 2: Diversity-Aware Data Acquisition (DADA).

Data: U pool de patches não rotulados, Q número de pontos a obter na presente iteração de aquisição, K número de clusters usado
Result: A patches obtidos; U

```
1  (PU, DF) ← MedidaIncerteza(U);
2  DFR ← ReduçãoDimensionalidade(DF);
3  CS ← Cluster(U,DFR,K);
4  CS ← OrdenarPorIncerteza(CS,PU);
5  for i ← 1 to K do
6  |   acqSize ← PR(CS_i) * Q;
7  |   A ← ObterPatches(CS_i, acqSize);
8  end
9  U ← U − A;
10 L_{i+1} ← L_i + A;
```

Na linha 3, o algoritmo agrupa os *patches* do *pool* (U) usando as *features* (DFR) produzidas após o processo do PCA. Na implementação realizada para os experimentos, essa etapa é feita usando Minibatch K-Means [164] para obter melhor eficiência de execução. Como resultado, são produzidos K *clusters* de *patches* (CS). Vale lembrar que no caso das implementações originais das abordagens por Ensembles e Bayesiana, descritas na Seção 4.1, não são usadas quaisquer medidas de densidade de informação para organizar os *patches* durante a aquisição, de forma que seriam retornados os Q *patches* de maior incertezas, indicados pelo vetor PU na linha 1.

A lógica de combinação é mostrada nas linhas 4 a 8. Esse processo se inicia pela ordenação dos *patches* em cada *cluster* de acordo com seus valores de incertezas (linha 4). Para cada *cluster* i, o número de *patches* a ser selecionado, $acqSize$, é calculado (linha 6) e aqueles com maior nível de incerteza são os escolhidos do *cluster* (CS_i). O valor de $acqSize$ para cada conjunto varia de acordo com a incerteza média dos *patches* que compõem o *cluster*.

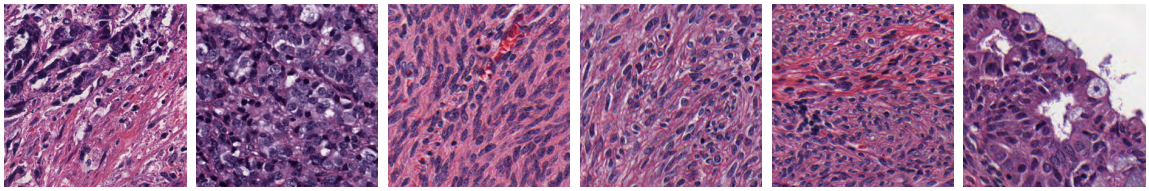
A razão para essa escolha se baseia no argumento que conjuntos compostos de imagens de maior incerteza devem trazer maior contribuição para o aprendizado e, portanto, maior número dessas imagens deve ser inserido no conjunto de treinamento. Assumindo que a média das incertezas dos exemplos de um conjunto i é $Z(i)$, o percentual de *patches* que cada *cluster* contribuirá para a aquisição será $PR(i) = Z(i)/\sum_K Z(i)$. O valor $PR(i)$ é então multiplicado por Q para se obter o número real de *patches* do *cluster* i que serão escolhidos para serem rotulados.

É importante destacar que nem sempre os *clusters* efetivamente possuem $PR(i) * Q$ *patches* para contribuir. Nesses casos, um *cluster* é aleatoriamente selecionado para fornecer mais *patches* para o conjunto de treinamento, de modo a se obter o número total de Q imagens solicitadas. A seleção aleatória foi escolhida, nesse caso, pois não se estabeleceu uma lógica de priorização entre *clusters* no sentido de se considerar que um determinado tipo de características pudesse ser mais relevante que outras.

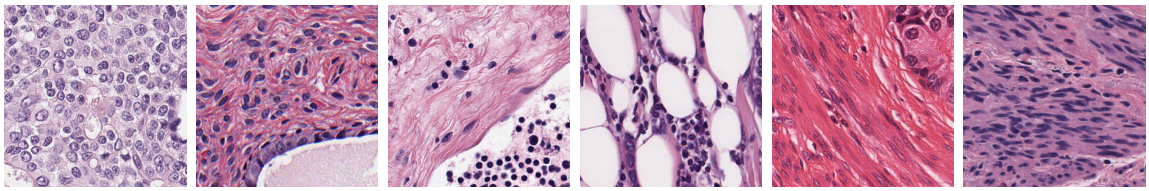
Dado que os *patches* são ordenados por incerteza em cada *cluster*, a função da linha 7 apenas retornará os primeiros $acqSize$ *patches*. Posteriormente, o *pool* U é atualizado para remover as imagens de A (linha 9) bem como o conjunto de treinamento da iteração seguinte recebe as imagens selecionadas na iteração atual (linha 10). A lógica de aquisição proposta estabelece uma relação entre incerteza e a heterogeneidade do espaço de características (ou diversidade de tecidos). Isso pode ser visto na Figura 6.2, onde os *patches* selecionados com e sem o uso do DADA são mostrados.

A Figura 6.2 traz uma visualização do que foi exposto no parágrafo anterior. Seis *patches* da aquisição de número 15 em um dos experimentos, selecionados pela abordagem utilizando Ensembles e BALD como métrica de incertezas, são apresentados na Figura 6.2a. Esses patches estão entre aqueles de maior incerteza de um *pool* de 100.000 imagens. Como pode ser observado, as imagens selecionadas por essa abordagem possuem características visuais muito semelhantes. Nas mesmas condições, o DADA fez uma seleção sobre essas imagens, parte delas apresentada na Figura 6.2b.

A heterogeneidade entre as imagens selecionadas pelo DADA se contrasta com a homogeneidade daquelas obtidas pela solução via Ensembles/BALD. Por outro lado, o CoreSet (Seção 4.1), que considera exclusivamente a diversidade dos dados, consegue abarcar a diversidade morfológica das imagens mas tem seu desempenho limitado, visto que os ní-



(a) Patches selecionados pela estratégia via Ensemble usando BALD como função de incertezas.



(b) Patches selecionados pelo DADA com BALD como função de incertezas na estratégia via Ensembles.

Figura 6.2: Painéis de *patches* adquiridos sem DADA (a) e pelo DADA (b). A seleção de *patches* ilustra a diversidade de características de tecidos selecionados usando DADA.

veis de incertezas também são importantes para a seleção de *patches* no AL, como será demonstrado no Capítulo 7.

6.1.1 DADA QuickAnnotator (DQA)

A utilização do DADA na prática depende da disponibilização de uma interface gráfica que seja simples para o usuário, ao mesmo tempo em que mantenha o fluxo de ações esperadas para a execução das anotações em uma sequência iterativa de seleção. Além disso, o usuário deve poder visualizar claramente quais *patches* devem receber anotações, bem como regiões de tecido em seu redor, para melhor contextualizar as características morfológicas e celulares presentes.

Com esse fim, optou-se por realizar adaptações a uma ferramenta já existente, denominada QuickAnnotator (QA) [165]. O QuickAnnotator implementa uma interface web que executa um fluxo de trabalho diferente do esperado para o DADA. No QA, está prevista a realização de revisões iterativas de anotações geradas automaticamente pela CNN, executadas por um profissional, semelhante ao ilustrado na Figura 5.1. O objetivo original do QA é produzir uma grande base de dados anotados reduzindo o custo de tempo de anotação, ao passo que o DADA busca produzir a menor base de dados possível que permita um modelo atingir um nível de desempenho de classificação desejado.

O DQA implementa um fluxo composto de três etapas:

- pré-processamento: um *pool* de *patches* deve ser gerado, a partir da extração dos mesmos dos WSIs desejados;

- inicialização: com a criação de um novo projeto e geração de conjunto de treinamento inicial; e
- iteração: anotação de um conjunto de *patches* já escolhidos e obtenção de um novo conjunto para anotação em um ciclo até que se atinja os objetivos para a geração da base de dados.

As iterações de anotações podem ser intercaladas por testes do último modelo treinado com os *patches* já anotados, para verificação do desempenho de classificação (Figura 6.3). Tanto um número de *patches* anotados quanto um determinado nível de desempenho do modelo podem ser usados como critério para que o usuário conclua as anotações de que necessita. O DQA permite a configuração de vários importantes parâmetros como: tamanho do conjunto inicial; o número de *patches* selecionados pelo AL a cada iteração; tamanho de *subpools* (Seção 6.2); uso do NAR (Seção 6.3.1); localização de WSIs e *pool* de *patches*, entre outros.

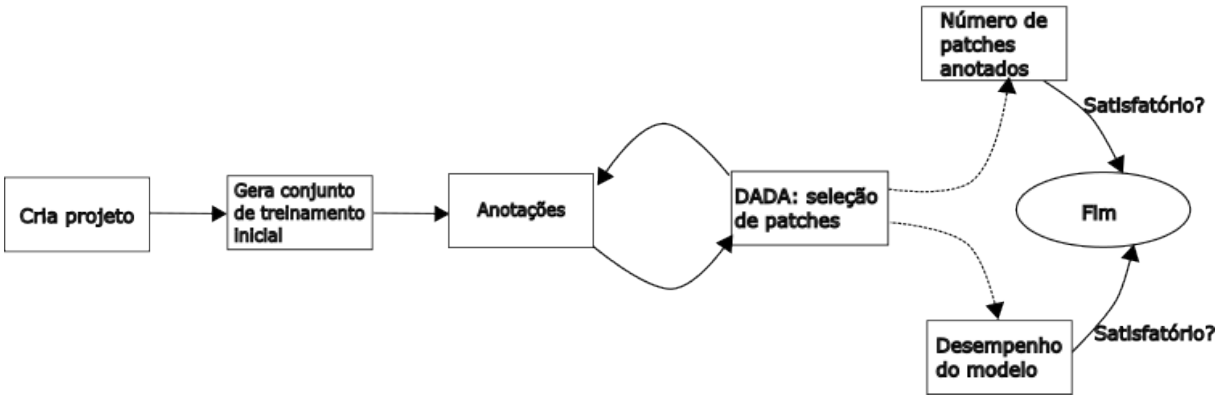
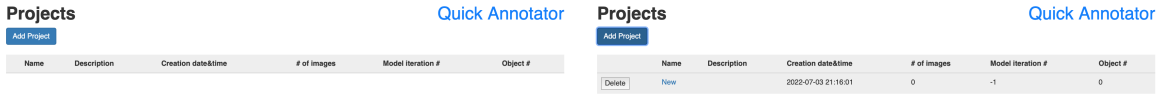


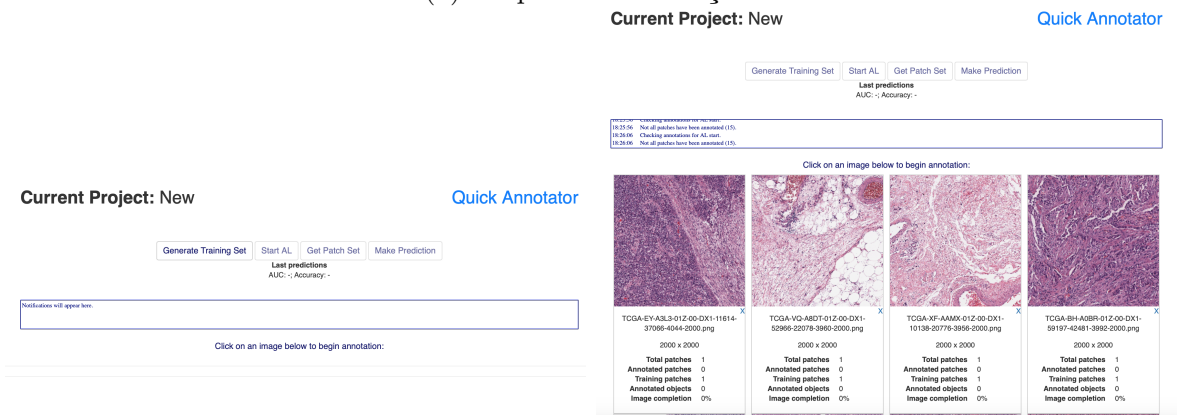
Figura 6.3: Fluxo de trabalho do DQA.

A Figura 6.4a ilustra a interface enquanto na etapa de inicialização, com a criação de um projeto de anotações específico. Após essa criação, o usuário pode solicitar a geração de um conjunto de treinamento inicial, sendo mostrados os *patches* contidos nesse conjunto (Figura 6.4b), os quais são aleatoriamente selecionados.

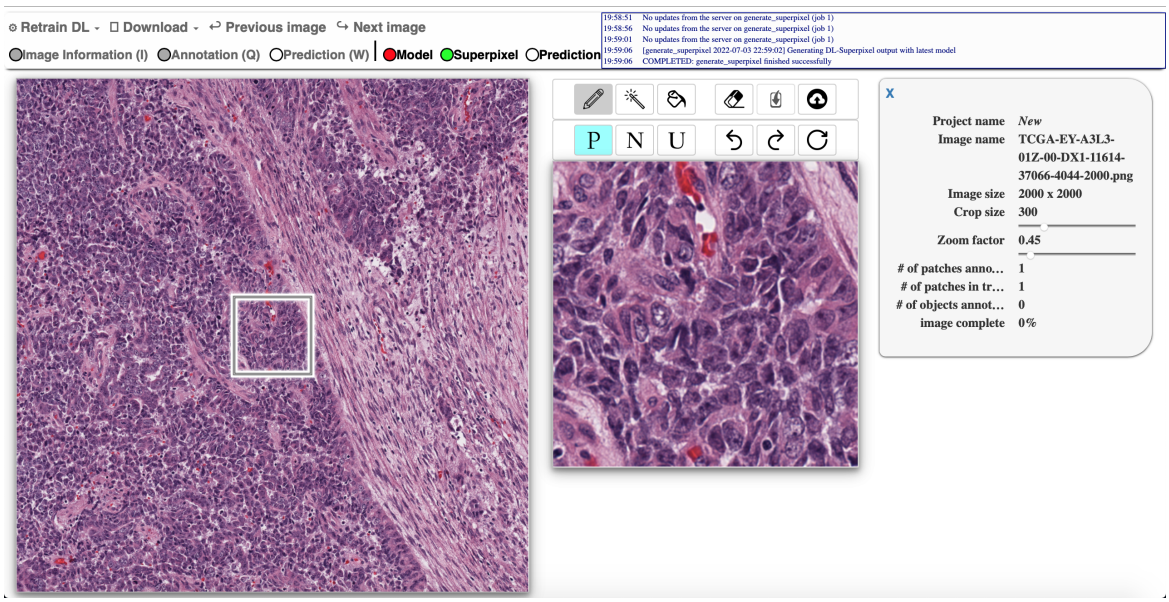
As anotações são iniciadas ao se clicar em uma das imagens miniaturizadas, que levará o usuário a uma nova tela (Figura 6.4c). Nessa nova tela serão mostrados dois quadros com imagens. À esquerda, a região ao redor do *patch* que deve receber anotação e uma região quadrada demarcando o *patch* em si. O quadro central mostra a região demarcada e ampliada para melhor visualização, bem como as ferramentas de anotação. No caso do DQA, o objetivo é atribuir uma classificação binária (positivo/negativo) para o *patch*



(a) Etapa de inicialização.



(b) Geração de conjunto de treinamento inicial e primeiro conjunto de *patches* para anotação.



(c) Interface de anotação de *patches*.

Figura 6.4: DADA QuickAnnotator. Interfaces gráficas.

como um todo, sendo necessário utilizar a ferramenta de preenchimento para a realização da anotação.

Após concluídas e salvas as anotações de todos os *patches* selecionados pelo DADA será possível solicitar um novo conjunto. Essa solicitação dá início à execução de uma nova

iteração, que vai treinar um novo modelo reduzido com o conjunto de imagens anotadas. Ao final desse processo, o usuário verá um novo conjunto de *patches* que devem receber rótulos. Importa destacar que, no momento em que o usuário solicitar o teste do modelo, será treinada a CNN alvo, não reduzida, que é a rede para a qual se pretende gerar um conjunto de treinamento.

6.2 Acelerando o DADA com *Subpools*

O principal objetivo do aprendizado ativo, no problema em foco, é maximizar o desempenho de classificação de um modelo utilizando, para isso, um conjunto de treinamento com tamanho mínimo. Entretanto, o uso dessa técnica em análise de imagens de patologia cria desafios em termos de custo computacional, devido à alta resolução das imagens e os grandes *pools* de dados usados. Isso resulta em cálculos de incertezas custosos, já que sua execução demanda passagens preditivas sobre todos os *patches* do *pool*, para todas as aquisições.

O alto custo computacional pode ser amenizado com a redução do tamanho dos *pools*, ao custo provável de perda de desempenho de classificação do modelo. Com um *pool* muito pequeno, o espaço de decisão do AL é reduzido, podendo não abranger o universo de imagens de forma representativa e degradar o aprendizado, impactando na capacidade de generalização dos modelos.

Esse problema pode ser abordado com o uso de *subpools* (S_U^i), que são subconjuntos do *pool* original (U^i), substituindo o último em cada etapa de aquisição do AL. Os conjuntos S_U^i são configurados para serem pequenos em relação ao conjunto inicial, mas são regenerados através da seleção aleatória de novos membros a partir de U^i a cada α etapas de aquisição do aprendizado. Essa estratégia mantém S_U^i pequeno, gerando uma redução expressiva do custo computacional (ver Seção 7.2.5), enquanto, ao mesmo tempo, a regeneração permite que esse *subpool* renove sua diversidade e, consequentemente, continue a ser um subconjunto representativo do conjunto inicial U^i.

Como mencionado, a regeneração ocorre a cada α etapas de aquisição. Um contador k acompanha quantas iterações i são realizadas até que o *subpool* seja regenerado. Durante essas iterações, os *patches* selecionados são acumulados em um conjunto A_α e, no momento da regeneração, é realizada a atualização do *pool* original, sendo removidos os elementos adquiridos durante essa geração, conforme a Equação 6.1.

$$
\begin{aligned}
k &= i \; mod \; \alpha; \\
A_\alpha &= \begin{cases} A_\alpha \cup \; A_i & \text{se k} > 0 \\ \emptyset & \text{se k} = 0, \end{cases} \\
S_U^{i+1} &= \begin{cases} S_U^i - A^i & \text{se k} > 0 \\ rand(U^{i+1}, \kappa) & \text{se k} = 0. \end{cases}
\end{aligned} \tag{6.1}
$$

A atualização do *pool* não rotulado e remoção dos *patches* adquiridos na geração atual ocorre a cada α iterações, ou seja, quando $k = 0 \rightarrow U^{i+1} = U^i - A_\alpha$. O *subpool*, por sua vez, é mantido atualizado a cada aquisição dentro de uma mesma geração, com a remoção dos pontos selecionados: $S_U^{i+1} = S_U^i - A^i$; e recriado a partir de uma seleção aleatória sobre o conjunto não rotulado, momento em que κ novos *patches* integrarão o *subpool*. Os itens contidos no *subpool* e não selecionados continuam presentes no *pool* original, podendo ser adquiridos em iterações futuras caso voltem a compor o *subpool*.

6.3 Acelerando o DADA Reduzindo Custos de Aquisição

O processo de seleção de novos *patches* para receberem anotações é complexo em dois aspectos: (i) em termos de desenvolvimento/escolha de um método que seja efetivo para o contexto da análise de imagens de patologia e (ii) com respeito ao tempo/poder computacional necessário para executar um método de AL. Percebeu-se, como será demonstrado nos resultados experimentais, que métodos tradicionais de AL, utilizados na literatura [117, 118] em outras aplicações, não apresentam boa performance quando usados em patologia digital.

Essa baixa performance se deve, com relação ao primeiro item, porque o tecido biológico possui características similares em grandes áreas, fazendo com que diversos *patches* com informações visuais muito semelhantes sejam selecionados para receber anotações. Assim, quando o método de AL ordena *patches*, por exemplo, a partir de seus níveis de incertezas, aqueles do topo, ou seja, de maior incerteza, se parecem muito ou possuem tecidos muito semelhantes. Ao ser adicionados ao conjunto de treinamento, esses *patches* contribuem muito suavemente ao processo de aprendizado mas geram impacto no tempo de treinamento dos modelos intermediários, influenciando também no segundo aspecto listado.

O principal ponto que afeta o custo computacional, gerando baixa performance dos modelos tradicionais em aplicações de patologia, se relaciona ao custo para se decidir quais *patches* devem ser incluídos entre os dados de treinamento. Esse custo elevado

se dá pois (i) usualmente o número de *patches* disponíveis nos conjuntos não rotulados é grande e deve ser calculado o nível de incerteza de cada um, tornando o processo caro, pois as metodologias usadas atualmente demandam a execução de múltiplos passos de classificação de todos os itens (por exemplo, número de passagens usado pelo MC-Dropout ou número de modelos em um Ensemble [117, 118]) e (ii) a cada iteração, os modelos intermediários devem ser treinados novamente utilizando todo o novo conjunto de treinamento. A utilização de *subpools* apresentada na seção anterior focou no primeiro desses dois fatores de custos, todavia, como se demonstrará no Capítulo 7, ainda não se mostrando uma solução suficiente.

Esses dois desafios técnicos são abordados pela solução de simplificação descrita nas seções a seguir, com os resultados demonstrados na Seção 7.2.6.

6.3.1 Custo de Incertezas e Treinamento de Modelos Intermediários

Cada iteração do AL pode ser conceitualmente dividia em duas etapas: a etapa de treinamento do modelo intermediário sobre o conjunto de treinamento L_i (ver Algoritmo 2) e a etapa de aquisição, em que o novo conjunto A_i é definido. No DADA, os modelos intermediários são utilizados para gerar incertezas e para fazer o agrupamento dos *patches*, a partir das *features* extraídas pela rede. A formulação original do DADA (e também das demais soluções de AL) utiliza a arquitetura para a qual se deseja gerar um conjunto de treinamento como modelo intermediário para realizar essas duas etapas. Essa arquitetura alvo é normalmente uma CNN complexa cuja qualidade de classificação para o domínio já é conhecida. A parcela de tempo correspondente a cada etapa é apresentada na Seção 7.2.5.

A estratégia de Modelos Surrogate (MS), em que modelos numéricos mais baratos (apesar de menos precisos) são usados para acelerar cada etapa ou iteração de uma aplicação, já foi utilizada com sucesso em outros domínios [166, 167]. Inspirados por esses modelos, uma estratégia similar é proposta, substituindo uma arquitetura CNN alvo por outra mais compacta, de menor custo computacional, para ser utilizada no ciclo de aprendizado ativo.

Um modelo mais simples em termos de custos computacionais reduziria o tempo total de cada iteração, visto que seu treinamento seria mais rápido e as inferências realizadas por ele durante a etapa de aquisição também seriam feitas com maior velocidade. A rede compacta, em tese, poderia ser escolhida meramente com base em seu tempo de execução, de forma a produzir reduções substanciais do custo de aquisição. Todavia, a qualidade da seleção produzida pelo DADA depende tanto das incertezas geradas quanto do agru-

pamento de *patches*, efetuado com base nas *features* extraídas pelas redes intermediárias, a cada iteração, de forma que o tempo de execução não pode ser o único fator levado em conta. Tanto o cálculo de incertezas quanto esse agrupamento seriam feitos a partir dos modelos de menor custo. Dessa forma, é preciso que a escolha desses modelos seja feita de maneira cautelosa e sua aplicação validada na prática.

É importante relembrar que o aprendizado ativo é uma técnica de otimização para produção de bases de dados de treinamento específicas para um determinado classificador. Assim, a utilização de uma rede mais barata para realização dos cálculos de incertezas e agrupamento de *patches* é uma extrapolação do conceito inicial. A fim de manter uma alta correlação entre a rede mais barata e a rede alvo, foram estudadas soluções de simplificação de CNNs, que geram versões de menor custo computacional de redes complexas e que pudessem manter a qualidade de seleção da rede original.

Apesar de implicar em uma extrapolação do conceito de aprendizado ativo, a utilização de redes intermediárias mais baratas se justifica pelo potencial ganho de tempo de processamento. A arquitetura utilizada nessas fases possui relevante impacto no tempo total necessário para o processo de aquisição de novos *patches* para anotação, visto que o número de parâmetros do modelo, entre outras variáveis, é diretamente proporcional ao número de operações matemáticas realizadas durante o treinamento e a inferência e, logo, no tempo de execução de cada iteração.

Uma métrica comumente utilizada para se avaliar o custo computacional de CNNs é a contagem de operações de ponto flutuante (FLOPs) realizadas pelo modelo durante uma inferência [168] (usualmente expressa em unidades na potência de 10^9 ou GFLOPs). Essa métrica, apesar de não corresponder a uma medida de tempo propriamente dita, permite comparar diferentes modelos e estimar qual deles apresentará um tempo de inferência e de treinamento menor.

O cálculo normalmente feito para a obtenção do número de FLOPs de uma rede considera a geração de uma previsão sobre uma imagem e é tido como uma característica da arquitetura, dado uma imagem de entrada de tamanho pré-definido. A contagem de FLOPs de uma CNN é usualmente feita a partir de uma aproximação que considera apenas as camadas convolucionais e totalmente conexas da rede, conforme a equação:

$$\begin{aligned} F_{conv} &= 2 * Número\ de\ canais * Dimensão\ do\ kernel * Dimensão\ de\ saída; \\ F_{denso} &= 2 * Dimensão\ de\ entrada * Dimensão\ de\ saída, \end{aligned} \tag{6.2}$$

em que F_{conv} corresponde ao número de operações realizadas em uma determinada camada convolucional da rede e F_{denso} às operações de eventuais camadas totalmente conexas. O número de FLOPs do modelo é, então, o somatório desses valores para todas as camadas

em questão.

Apesar de se referir ao momento de inferência, a contagem de FLOPs também permite estimar o custo de treinamento da rede. Considerando a aplicação de uma das variantes do algoritmo de *backpropagation* [66] como processo de otimização durante o treinamento, o custo total é dado pela somatório do custo de inferência (da etapa *forward*) e do custo da fase de *backpropagation*. De maneira simplificada, na primeira etapa é realizada a inferência sobre um *batch* de entrada e obtido o custo de desvio através da função de custo, enquanto que na segunda ocorre o cálculo dos gradientes de cada peso para todas as camadas e a atualização dos pesos, conforme a taxa de aprendizado e os gradientes obtidos.

O cálculo do custo computacional durante o treinamento sofre influência de variáveis que não são consideradas durante as inferências, como: tamanho do *batch* de entrada e número de operações requeridas para o cálculo de gradientes [131, 169]. Em experimentos empíricos, já foi sugerido que há uma razão aproximada entre o número de FLOPs que um modelo demanda durante inferência e a demanda durante o treinamento [170], com indicações de que essa razão varia entre 1:1 até 1:3, sendo 1:2 uma estimativa correspondente à maior parte dos modelos e *batches* de entrada usados na prática. Assim, a partir da redução do número de FLOPs da rede, espera-se uma diminuição tanto do tempo de inferência quanto de treinamento, contribuindo duplamente para uma aceleração do processo de aquisição de novas imagens de anotação do DADA.

Dessa forma, os altos custos computacionais da utilização de modelos de *deep learning* em aprendizado ativo podem, em tese, ser abordados pela simplificação de CNNs, que se mostra uma solução atrativa para a aplicação de AL. A simplificação pode se dar através de diferentes métodos, como: a poda (*pruning*) [171, 172, 173], esparsificação (*sparsification*) [174, 175], quantização (*quantization*) [176, 177], entre outras. Entre esses métodos, as soluções de poda são as que receberam maior atenção da literatura, com a maioria das abordagens se concentrando em remover filtros nas camadas convolucionais, referidas como poda de canais ou de filtros [129, 178, 135]. Outras soluções atuam em uma gama maior de estruturas, removendo camadas ou até blocos de camadas [140].

Soluções de poda de redes neurais já foram o foco de inúmeras publicações (ver Seção 4.2), apresentando bons resultados na aceleração do tempo de processamento demandado por CNNs e, inclusive, com soluções sem perdas de desempenho de classificação [141]. Retomando aquilo já exposto nessa seção e na Seção 4.2, ao se aplicar qualquer processo de poda, a rede reduzida resultante será diferente da rede alvo para a qual se deseja selecionar as melhores imagens para se receber anotações. Essa diferença traz um problema teórico, uma vez que a Equação 3.14, aplicada para o cálculo das incertezas de cada imagem, claramente é dependente de um modelo específico, visto que a aproximação

entre a probabilidade posterior e a distribuição de pesos produzidos pelo *dropout* ou por Ensembles é específica para um dado conjunto de pesos: $p(\omega|X,Y) \approx q_\theta(\omega)$.

Assim, na prática, o cálculo de incertezas realizado por uma rede simplificada de uma rede alvo implica na transferência dessas incertezas entre as duas redes envolvidas. Essa transferência pode resultar em perdas de desempenho de classificação da rede alvo, uma vez que a rede simplificada pode não capturar todas as características das imagens não rotuladas e a transferência de incertezas entre os modelos pode não ser eficaz. A solução de simplificação desenvolvida neste trabalho é apresentada a seguir e leva em consideração esse impedimento teórico, bem como procura sanar outras dificuldades.

6.3.2 Network Auto-Reduction (NAR)

Os objetivos em desenvolver um método de simplificação de redes específico para o DADA foram: primordialmente que a rede reduzida apresentasse uma menor contagem de FLOPs possível, reduzindo tanto os custos de treinamento de modelos intermediários quanto de geração de incertezas; produzir uma versão simplificada que mantivesse uma qualidade de classificação próxima à da original, na hipótese de que o agrupamento de *patches* realizado pela rede simplificada fosse capaz de selecionar imagens significativas para a rede alvo; e manter a estrutura da rede reduzida próxima da rede original, considerando a hipótese de que essa semelhança manteria também uma correlação entre as incertezas produzidas pelos modelos.

Assim, para que esses objetivos pudessem ser atingidos, especialmente com relação à redução do custo computacional, o método de redução deve introduzir o mínimo de *overhead* possível para o tempo de cada iteração do aprendizado ativo. É importante destacar que o objetivo dos métodos de simplificação disponíveis na literatura (ver Seção 4.2) normalmente é minimizar a contagem de FLOPs do modelo reduzido e, ao mesmo tempo, manter o desempenho de classificação o mais próximo possível do apresentado pelo modelo original. Apesar de considerarmos que a manutenção do desempenho de classificação seja diretamente relacionada com a qualidade do agrupamento de *patches* efetuado pelo DADA, esse não é o propósito do NAR.

As abordagens de simplificação de redes já apresentadas na Seção 4.2 buscam reduzir uma das dimensões de uma CNN: profundidade, largura ou resolução [141]. Alguns dos estudos propuseram a remoção de filtros convolucionais específicos [128, 134, 135, 179, 136], a introdução de esparsamento de pesos [173] ou uma combinação de ambos [140, 141]. Essas soluções, todavia, não produzem reduções de FLOPs com a intensidade procurada.

Por outro lado, a poda de filtros ou de estruturas da rede em geral é abordada por várias soluções que oferecem possibilidades de se escolher de quais camadas devem ocorrer as remoções ou quais as estruturas devem ser excluídas, de maneira a gerar o menor

impacto de classificação possível. Apesar disso, esse processo não é realizado de uma maneira balanceada, considerando todas as dimensões do modelo de maneira integrada, o que pode limitar o desempenho e a acurácia do modelo reduzido [134, 180, 181]. Adicionalmente, os processos de escolha das estruturas a serem excluídas demandam a execução de procedimentos caros, como treinamentos das redes ou a inserção de estruturas e/ou camadas específicas para o cálculo de parâmetros [141].

Procurando levar em conta tanto o aspecto de qualidade de seleção de *patches* quanto os ganhos em tempo de execução, a proposta desenvolvida no presente trabalho automaticamente cria uma CNN simplificada a partir de um modelo alvo, através de um processo aqui denominado NAR. O NAR produz uma rede reduzida de um modelo alvo de maneira controlada, através da exclusão de camadas inteiras (redução de profundidade), de filtros convolucionais (redução de largura) e da redução da resolução de tensores. Dessa forma, busca-se produzir uma rede com menor consumo de recursos mas que mantenha propriedades da CNN alvo. O processo de simplificação desenvolvido pelo NAR é mostrado na Figura 6.5.

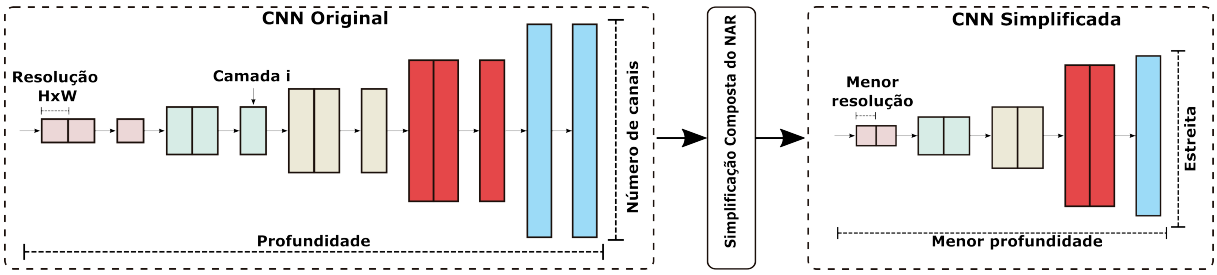


Figura 6.5: A simplificação composta do NAR modifica profundidade, largura e resolução de entrada para se obter um balanceamento entre os componentes da CNN.

O NAR se inspira na abordagem proposta por Tan et al. [182] para expandir redes simples. O método por eles proposto foi projetado para aumentar uma CNN, a fim de melhorar o desempenho de classificação, ao mesmo tempo em que busca utilizar a menor quantidade possível de recursos computacionais para isso. No contexto aqui trabalhado, contudo, busca-se simplificar uma CNN que sabidamente já possui boa capacidade de classificação no domínio sob foco mas que tem um custo muito alto. Tan et al. [182] formularam o problema de expandir uma CNN como um problema de otimização, definido

na Equação 6.3, dado objetivo de consumo de memória (TM) e contagem de FLOPs (TF):

$$\begin{aligned} \max_{d,w,r} \quad & Accuracy(\mathbb{M}(d,w,r)) \\ s.t \quad & \mathbb{M}(d,w,r) \;=\; \bigodot_{i=1,\ldots,s} \hat{\mathcal{F}}_i^{d.\hat{L}_i}\left(X_{\langle r.\hat{H}^i, r.\hat{W}^i, w.\hat{C}^i\rangle}\right) \\ Memory(\mathbb{M}) \;&\leq\; TM; \\ FLOPs(\mathbb{M}) \;&\leq\; TF, \end{aligned} \tag{6.3}$$

Na equação, $\bigodot_{i=1,\ldots,s}$ representa a composição das camadas de uma dada CNN $\mathbb{M}$. A formulação apresentada produzirá arquiteturas de redes estruturadas em blocos de $d.\hat{L}_i$ ocorrências de uma determinada configuração de camada, as quais se repetem em uma sequência. Cada camada $\hat{L}_i$ pode ser compreendida como a aplicação da função $\hat{\mathcal{F}}_i$ em seu tensor de entrada X_i, com dimensões $\hat{H}^i, \hat{W}^i, \hat{C}^i$ (altura, largura, canais).

Assim, a formulação original de extensão de redes modifica todas as três dimensões do modelo simultaneamente e de uma maneira balanceada:

- *profundidade*: d ocorrências da camada $\hat{L}_i$ em um determinado bloco;
- *largura*: cada nova camada contendo $w.\hat{C}_i$ canais; e
- *resolução*: a altura $\hat{H}_i$ e largura $\hat{W}_i$ do tensor X_i são ajustados por um fator de w, com exceção da camada $i = 0$, caso em que as dimensões de entrada são as mesmas que as dimensões da imagem de entrada e seus canais de cores.

Os coeficientes de escalonamento d, w, r usados por Tan et al. permitem criar um modelo mais complexo $\mathbb{M}$, cujo incremento em custo computacional depende desses coeficientes. Para as camadas convolucionais, considerando a formulação para o cálculo de FLOPs dada pela Equação 6.2, com $d.\hat{L}_i$ ocorrências da camada i e tamanho de saída correspondente a $\hat{H}_{\mathcal{M}}^{i+1} = r.\hat{H}^{i+1}, \hat{W}_{\mathcal{M}}^{i+1} = r.\hat{W}^{i+1}, \hat{C}_{\mathcal{M}}^{i+1} = w.\hat{C}^{i+1}$, o custo da CNN estendida é incrementado de maneira proporcional a $d.w^2.r^2$.

De acordo com Tan et al., o balanceamento dos coeficientes de escalonamento é crucial para se obter a melhor relação entre acurácia/custo para um dado conjunto de restrições de recursos. Para isso, uma estratégia de ajuste de tamanho uniforme é utilizado, para que o aumento dos custos seja distribuído entre todos os parâmetros, através de um coeficiente ϕ, conforme Equação 6.4.

$$
\begin{aligned}
d &= \alpha^{\phi} \\
w &= \beta^{\phi} \\
r &= \gamma^{\phi} \\
s.t \;\; & \alpha.\beta^2.\gamma^2 \approx 2, e \\
& \alpha \geq 1, \beta \geq 1, \gamma \geq 1,
\end{aligned}
\tag{6.4}
$$

considerando valores de α, β e γ pré-determinados. Em seu trabalho, Tan et al. aplicaram *grid search* para chegar a um conjunto de valores apropriado para o modelo base utilizado, denominado EfficientNet [182]. Nesse caso, os valores encontrados ao fixar $\phi = 1$ e executar a busca foi de $\alpha = 1,2; \beta = 1,1; \gamma = 1,15$. Esses valores, apesar de serem específicos para o passo entre a rede base $\phi = 0$ e a primeira evolução com $\phi = 1$, foram utilizados pelos autores em passos de incremento seguintes, visto que a busca utilizando modelos mais complexos se torna inviável. Adicionalmente, esses mesmos valores ainda foram utilizados sobre arquiteturas distintas para demonstração do método de expansão de redes por eles proposto, sendo aqui utilizados também como referência. É importante destacar que ao se impor a restrição que $\alpha.\beta^2.\gamma^2 \approx 2$, o aumento esperado do custo de uma CNN, medido em número de FLOPs, é de 2^{ϕ} em relação ao modelo base.

A modelagem produzida para a expansão de redes, até aqui apresentada, apesar de ter um objetivo contrário ao desejado no presente trabalho, embasa a construção do NAR. Com o intuito de manter o balanceamento entre as reduções de custo computacional entre as dimensões da rede e, buscando manter a melhor relação entre acurácia/custo, são aplicados coeficientes de redução conforme a Equação 6.5. Mantendo-se as mesmas restrições aos valores de α, β e γ, partindo-se de um modelo complexo ($\phi = 0$), espera-se uma redução da contagem de FLOPs de $\frac{1}{2^{\phi}}$ para cada grau de redução aplicado pelo parâmetro ϕ.

$$
\begin{aligned}
d &= \alpha^{-\boldsymbol{\phi}}; \\
w &= \beta^{-\boldsymbol{\phi}}; \\
r &= \gamma^{-\boldsymbol{\phi}}; \\
s.t \;\; & \alpha.\beta^2.\gamma^2 \approx 2, \\
& \alpha \geq 1, \beta \geq 1, \gamma \geq 1, and \\
& \phi \geq 1,
\end{aligned}
\tag{6.5}
$$

Os fatores de redução, conforme expresso na Equação 6.5, são tais que $0 < d, w, r < 1$, e sua aplicação se dá sobre as estruturas de CNNs com arquiteturas baseadas em blocos. Assim, a dimensão de profundidade é alterada ao se aplicar o coeficiente d sobre a quantidade de ocorrências de cada bloco, diminuindo sua repetição e, consequentemente,

o número de camadas final de uma CNN. A largura é ajustada aplicando-se o coeficiente w sobre o número de canais de cada camada convolucional e o coeficiente r diminui a resolução ao alterar as dimensões da imagem de entrada da rede, o que é propagado pelas demais camadas no momento em que as dimensões de saída de cada uma são definidas.

A ResNet50 V2 é uma arquitetura baseada em blocos, que serve de modelo base para uma família de CNNs mais complexas, sendo utilizada como modelo de demonstração de várias das técnicas de simplificação apresentadas na Seção 4.2. A Tabela 6.1 apresenta a estrutura dessa rede, bem como o resultado da aplicação do NAR sobre ela, com diferentes níveis de redução, considerando os valores de α, β e γ de referência conforme expresso anteriormente. Se destaca dessa tabela os efeitos da aplicação do NAR tanto sobre a arquitetura dos modelos reduzidos quanto sobre o número de parâmetros da rede obtida.

A ResNet50 V2 original possui 4 estágios, compostos por blocos de camadas que se repetem 3×, 4×, 6× e 3×, respectivamente, para cada estágio. No caso de uma redução ao primeiro nível ($\phi = 1$) a estrutura dos estágios é diferente, respectivamente com sequências de 2×, 3×, 5× e 2× blocos em cada. Combinado com a redução da largura de cada camada, a simplificação trazida pelo NAR implica em uma drástica redução do número de parâmetros do modelo, de 23.568.898 na versão original, para 8.514.988 no modelo reduzido com nível $\phi = 3$. A Seção 7.2.6 apresenta os resultados obtidos com o uso dos modelos reduzidos de diferentes CNNs com o DADA.

CNN	ORIGINAL	NAR $\phi = 1$	NAR $\phi = 2$	NAR $\phi = 3$
ResNet50 V2 [142] # Params	23,568,898	14,583,140	11,274,413	8,514,988
Conv 1	7 x 7, 64, stride 2			
Estágio 1	[1x1, 64 3x3, 64 1x1, 256] x3	[1x1, 58 3x3, 58 1x1, 232] x2	[1x1, 53 3x3, 53 1x1, 212] x2	[1x1, 48 3x3, 48 1x1, 192] x2
Estágio 2	[1x1, 128 3x3, 128 1x1, 512] x4	[1x1, 106 3x3, 106 1x1, 424] x3	[1x1, 116 3x3, 116 1x1, 464] x3	[1x1, 96 3x3, 96 1x1, 384] x2
Estágio 3	[1x1, 256 3x3, 256 1x1, 1024] x6	[1x1, 233 3x3, 233 1x1, 932] x5	[1x1, 212 3x3, 212 1x1, 848] x4	[1x1, 192 3x3, 192 1x1, 768] x3
Estágio 4	[1x1, 512 3x3, 512 1x1, 2048] x3	[1x1, 465 3x3, 465 1x1, 1860] x2	[1x1, 423 3x3, 423 1x1, 1692] x2	[1x1, 385 3x3, 385 1x1, 1540] x2

Tabela 6.1: Número de parâmetros e organização de camadas na ResNet50 V2 original e suas versões reduzidas pelo NAR. A contagem de parâmetros considera um problema de classificação binária. O número de filtros em cada camada é representado junto com a dimensão do kernel (p.ex. 1x1, 64 indica um kernel 1x1 com 64 filtros) e o número de repetições de cada bloco está expresso à direita das especificações das camadas.

A rede Inception V4 [183] já foi utilizada com sucesso em aplicações de histopatologia [163], sendo uma rede sabidamente eficaz para o problema de classificação de regiões de LIT em tecidos de biópsias. A aplicação do NAR em modelos mais complexos, como a Inception V4, se dá de maneira equivalente ao realizado sobre a ResNet50 V2. A rede InceptionV4 é significativamente mais profunda que a ResNet50, não sendo possível detalhar sua organização em blocos como feito anteriormente. Todavia, alguns dados das versões original e reduzidas são mostradas na Tabela 6.2. A partir dessa tabela, é possível verificar as especificações das dimensões profundidade e resolução de entrada e, juntamente com o reflexo da diminuição da largura sobre as camadas convolucionais, o efeito sobre o número de parâmetros do modelo.

CNN	**# Params**	**Camadas de pesos**	**# de camadas**	**Dimensão de entrada**
Inception V4 [183]	54.385.570	245	861	240x240
$\phi = 1$	38.603.526	206	723	209x209
$\phi = 2$	28.118.858	179	627	181x181
$\phi = 3$	20.002.127	145	507	158x158
$\phi = 4$	14.506.100	123	429	137x137
$\phi = 5$	10.275.579	101	351	119x119
$\phi = 6$	7.484.162	91	315	104x104

Tabela 6.2: Número de parâmetros e camadas na Inception V4 original (classificação binária) e suas versões reduzidas.

Nem todas as redes permitem os mesmos níveis de redução, visto que a estrutura de blocos e o número total de camadas apresenta diferenças significativas entre os modelos. A depender do valor ϕ e da quantidade de repetições de determinado bloco da rede, o coeficiente d pode se aproximar de zero, o que implicaria em uma ruptura do modelo. Adicionalmente, a fim de manter a estrutura de blocos do modelo reduzido e maior correlação com o modelo original, limitamos as reduções de forma que cada bloco tenha ao menos duas repetições. Essa limitação faz com que a ResNet50 V2, por exemplo, só alcance o nível $\phi = 3$, pois o coeficiente $\phi = 4$ exigiria que os estágios compostos originalmente por 3 blocos reduzissem essa quantidade para 1 bloco, o que tornaria esses estágios meramente sequenciais, desconfigurando a estrutura em blocos.

Frisa-se que, apesar da metodologia de simplificação do NAR ter foco inicial nas CNNs com estruturas em blocos, é possível aplicá-la também a redes puramente sequenciais, como é o caso da VGG16 [68] e demais modelos derivados. Para isso, é necessário alterar a forma de aplicação da redução da profundidade, inserindo ou não determinadas camadas conforme o nível ϕ definido pelo usuário. As demais dimensões da rede não demandam mudanças para sua aplicação.

Considerando os objetivos propostos para o desenvolvimento do NAR, é importante destacar que seu uso em conjunto com o DADA não traz qualquer novo *overhead* ao

processo de aquisição de *patches*, uma vez que o modelo reduzido é construído tendo por base unicamente o valor do parâmetro ϕ. Assim, espera-se obter uma significativa redução do tempo de aquisição, reduzindo a latência do processo de anotação, ao mesmo tempo em que o custo de processamento é diminuído. A transferência de incertezas tem sua eficácia avaliada experimentalmente, com resultados abordados na Seção 7.2.6, juntamente com a minimização do número de FLOPs das redes resultantes e os ganhos em tempo de execução.

Capítulo 7

Resultados

O DADA foi avaliado experimentalmente em duas tarefas de classificação. A primeira é a identificação de dígitos numéricos escritos à mão, contidos na base de dados MNIST [73], comumente utilizada em pesquisas sobre aprendizado ativo. A segunda tarefa corresponde à aplicação motivadora, descrita no Capítulo 5, que é a identificação de regiões em WSIs com presença de linfócitos infiltrantes de tumor (LITs).

A base de dados, nesse último caso, foi criada a partir de um conjunto de 62 WSIs, com a respectiva classificação de seus *patches* com respeito à presença de LITs. Essa classificação advém do trabalho realizado em [143], conforme descrito no Capítulo 5. Todos esses slides são disponibilizados para a comunidade acadêmica pelo *The Cancer Genome Atlas (TCGA)* [184].

Os slides utilizados para a geração dessa base de dados correspondem a 10 tipos de tecidos (Tabela 7.1), incluindo mama, próstata e casos de câncer pancreático. A quantidade de slides de cada tipo de tecido é detalhada pela Tabela 7.2.

Acrônimo	Tipo de Tumor
BLCA	Bladder urothelial carcinoma
BRCA	Breast invasive carcinoma
CESC	Cervical squamous cell carcinoma and endocervical adenocarcinoma
COAD	Colon adenocarcinoma
PAAD	Pancreatic adenocarcinoma
PRAD	Prostate adenocarcinoma
READ	Rectum adenocarcinoma
SKCM	Skin Cutaneous Melanoma
STAD	Stomach adenocarcinoma
UCEC	Uterine Corpus Endometrial Carcinoma

Tabela 7.1: Tipos de tumor.

Do total de 62 WSIs, 5 foram aleatoriamente selecionados para fornecer *patches* para um conjunto de testes, enquanto os 57 slides restantes forneceram *patches* exclusivamente

	prad	ucec	blca	stad	skcm	coad	brca	cesc	paad	read
Treino/Val	5	13	3	9	8	4	9	5	0	1
Teste	0	0	1	0	2	0	0	1	1	0
Total	5	13	4	9	10	4	9	6	1	1

Tabela 7.2: Tipos de tecidos e número de slides para cada conjunto.

para o conjunto de treinamento/validação dos modelos. O conjunto de testes efetivamente utilizado nos experimentos foi mantido constante, compreendendo 15.000 *patches* selecionados aleatoriamente entre aqueles originados dos slides reservados para esse fim. Os *patches* disponíveis para o treinamento/validação, originários dos 57 slides mencionados, totalizam 499.055 imagens, que são selecionadas aleatoriamente para formar o estoque de dados utilizados para o aprendizado ativo, conforme a configuração de cada experimento e as abordagens avaliadas.

Os IDs dos slides que compõem os conjuntos de treinamento/validação são relacionados na Tabela 7.3, enquanto aqueles que formam o conjunto de testes estão relacionados na Tabela 7.4. As configurações dos experimentos para as bases MNIST e de LITs são detalhadas na Tabela 7.5, considerando a utilização de um estoque de dados não rotulados de 100.000 *patches*, selecionados aleatoriamente do total daqueles disponíveis, para os experimentos que não utilizam a solução de *subpools* dinâmicos (ver Seção 6.2).

TCGA-EO-A22S-01Z-00	TCGA-FI-A2EY-01Z-00	TCGA-S5-AA26-01Z-00	TCGA-C5-A7CK-01Z-00	TCGA-EY-A3L3-01Z-00
TCGA-HG-A9SC-01Z-00	TCGA-EY-A2OO-01Z-00	TCGA-BS-A0UF-01Z-00	TCGA-BH-A202-01Z-00	TCGA-D7-6519-01Z-00
TCGA-AP-A0LH-01Z-00	TCGA-OD-A75X-06Z-00	TCGA-BG-A0W1-01Z-00	TCGA-BR-6709-01Z-00	TCGA-AP-A1DM-01Z-00
TCGA-BG-A0MQ-01Z-00	TCGA-W3-AA1V-01Z-00	TCGA-D3-A8GJ-06Z-00	TCGA-VQ-A8P8-01Z-00	TCGA-MY-A913-01Z-00
TCGA-XF-AAMX-01Z-00	TCGA-EE-A3AF-01Z-00	TCGA-D9-A149-01Z-00	TCGA-DS-A7WF-01Z-00	TCGA-EW-A1J1-01Z-00
TCGA-D1-A16N-01Z-00	TCGA-VQ-A94U-01Z-00	TCGA-VQ-A8DT-01Z-00	TCGA-EJ-5497-01Z-00	TCGA-BG-A0M4-01Z-00
TCGA-BR-8058-01Z-00	TCGA-C8-A12K-01Z-00	TCGA-D5-6534-01Z-00	TCGA-YL-A8HL-01Z-00	TCGA-EW-A1PF-01Z-00
TCGA-D1-A176-01Z-00	TCGA-F5-6864-01Z-00	TCGA-BR-8682-01Z-00	TCGA-HC-7078-01Z-00	TCGA-AM-5820-01Z-00
TCGA-EY-A549-01Z-00	TCGA-D8-A1XD-01Z-00	TCGA-XV-AAZW-01Z-00	TCGA-HC-8264-01Z-00	TCGA-EE-A2GS-01Z-00
TCGA-AU-6004-01Z-00	TCGA-FR-A7U8-01Z-00	TCGA-HU-A4GP-01Z-00	TCGA-HU-8610-01Z-00	TCGA-VS-A8EJ-01Z-00
TCGA-AO-A0JI-01Z-00	TCGA-EJ-5530-01Z-00	TCGA-E9-A3X8-01Z-00	TCGA-BH-A0BR-01Z-00	TCGA-AY-6196-01Z-00
TCGA-D8-A1XU-01Z-00	TCGA-C4-A0F6-01Z-00			

Tabela 7.3: IDs TCGA de cada slide para o conjunto de treinamento/validação.

TCGA-BL-A13J-01Z-00	TCGA-FR-A728-01Z-00	TCGA-EE-A2MH-01Z-00	TCGA-C5-A1MH-01Z-00	TCGA-US-A77G-01Z-00

Tabela 7.4: IDs TCGA de cada slide para o conjunto de testes.

Os nomes das principais abordagens avaliadas nos experimentos são detalhados na Tabela 7.6, sendo que as soluções propostas nesse trabalho são aquelas com o prefixo *DD*, usado nas demais tabelas e figuras do presente documento. Os modelos de *deep learning* foram treinados e testados em uma máquina com 2 GPUs NVIDIA V100 e 16 GB de memória dedicada em cada placa, além de 2 processadores Intel Xeon Gold 6148. O sistema operacional era Linux e os tempos de execução foram limitados a 40 horas, devido a restrições de configuração do sistema. As estratégias foram implementadas utilizando

os *frameworks* Keras e Tensorflow. Importante destacar também que foram adaptados códigos fornecidos por [118] para o cálculo das incertezas.

Dataset	**Modelo**	**Treinamento épocas**	**T/E**	**# Patches pool/val/teste**	**Aquisição tamanho**	**Total adquiridos**
MNIST	Keras + 2	50	50/3	59,900/100/9,100	20	1,000
TIL	InceptionV4	50	20/3	100,000/100/15,000	200	3,100 - 3,500

Tabela 7.5: Configuração de experimentos. Keras+2 é um modelo CNN modificado originário do Keras. T e E correspondem a passadas sobre os dados nas soluções MC Dropout e modelos no ensemble, respectivamente.

	MC Dropout		**Ensemble**	
Método	**Variation Ratios**	**BALD**	**Variation Ratios**	**BALD**
Versões do DADA	MC DDVR	MC DDBALD	ENS DDVR	ENS DDBALD
Versões originais	MC VR	MC BALD	ENS VR	ENS BALD

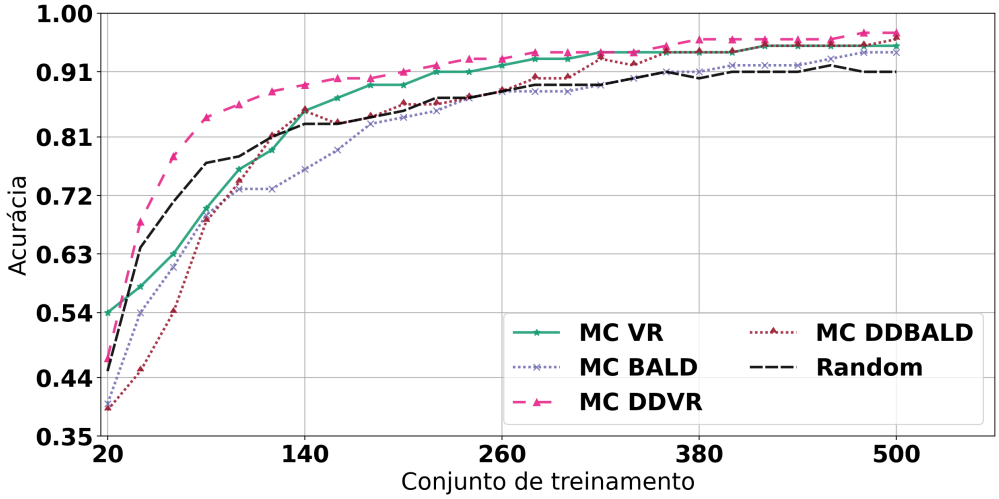
Tabela 7.6: Acrônimos usados para as estratégias (MC Dropout/Ensemble) e métrica de incertezas (VR/BALD).

A seguir são apresentados os resultados obtidos pelo uso do DADA, com uma etapa inicial de validação da solução sobre a base MNIST, seguida de comparações com as versões originais da seleção de *patches* via Ensembles, MC Dropout e a abordagem *CoreSet*. A seleção aleatória de *patches* (*Random*) é apresentada nesses primeiros experimentos como um *baseline* sobre o qual toda estratégia de aprendizado ativo deveria melhorar.

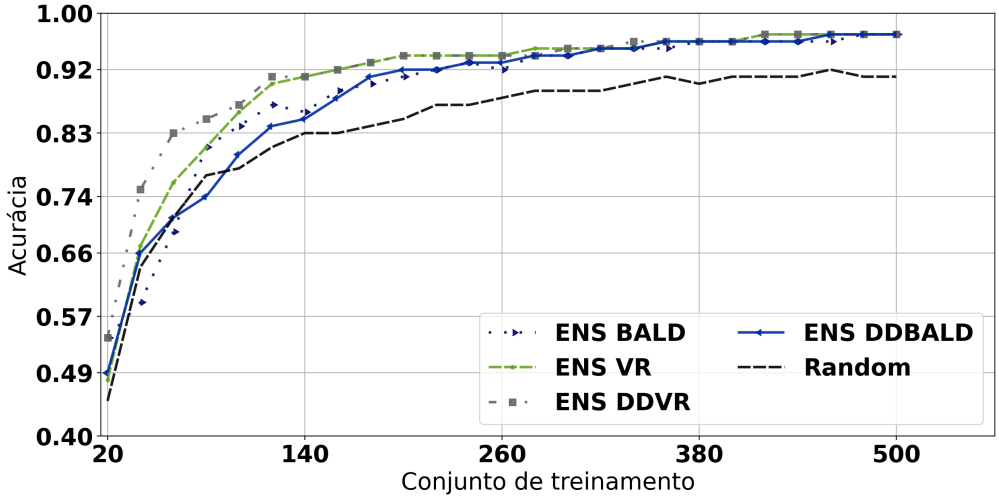
7.1 Base MNIST

A classificação sobre a base de dados MNIST utiliza um modelo CNN com 4 camadas convolucionais e ReLU como função de ativação (Keras+2), extendendo o modelo usado como exemplo pelo Keras [185], o qual possui 2 camadas convolucionais, para prover maior acurácia (Tabela 7.5). Dropout e camadas *max pooling* foram adicionados a cada 2 camadas convolucionais. Uma camada densa, uma camada de dropout e um *softmax* de 10 canais como função de ativação formam o topo do modelo, ou seja, compõem a porção de classificação.

As características para agrupamento dos *patches*, no caso do DADA, foram extraídas da saída da última camada convolucional e possuem 7.744 dimensões. O treinamento foi executado com o otimizador *Adadelta*, com taxa de aprendizado inicial de 0.0005, reduzida a 1/10 após 30 épocas, e tamanho do *batch* de 128 imagens.



(a) Estratégias utilizando **MC Dropout**



(b) Estratégias utilizando **Ensembles**.

Figura 7.1: Acurácia vs. # de imagens adquiridas. Ensemble (ENS) e Bayesian (MC) são avaliadas usando DADA (prefixo DD) ou não.

A Figura 7.1 apresenta os resultados com VR e BALD como funções de incertezas para as abordagens de AL por Ensembles (ENS) e Bayesiana (MC), tanto utilizando quanto não utilizando DADA. As estratégias melhoram o resultado obtido por uma seleção aleatória de maneira significativa para a maioria dos casos. A solução Bayesiana é superada pelos Ensembles quando a mesma métrica de incertezas é usada, o que corrobora resultados anteriores, obtidos por [117]. Ainda, a solução Bayesiana requer mais aquisições para superar a seleção aleatória, com a métrica VR demonstrando os melhores resultados.

		MC Dropout				Ensemble			
Acurácia	**Random**	**VR**	**BALD**	**DD VR**	**DD BALD**	**VR**	**BALD**	**DD VR**	**DD BALD**
80%	120	120	160	80	120	80	80	**60**	100
85%	200	140	220	100	140	100	120	**80**	140
90%	340	220	340	160	280	**120**	180	**120**	180

Tabela 7.7: Tamanho do conjunto de treinamento e acurácia nos níveis de 80%, 85% e 90%.

O uso do DADA com a solução Bayesiana (MC DDVR e MC DDBALD) resultou em um salto de desempenho quando comparados às versões sem o uso do DADA (MC VR e MC BALD). Para atingir uma acurácia de 85%, MC DDVR e MC DDBALD demandaram 100 e 140 imagens respectivamente, enquanto MC VR e MC BALD precisaram de 140 e 220 imagens (ver Tabela 7.7). A versão que obteve o melhor desempenho entre todas as demais foi o DADA via Ensembles, utilizando VR como métrica de incertezas.

Em um experimento demonstrando a eficácia do aprendizado ativo, um modelo foi treinado sobre uma base de dados de 59.500 imagens, com 500 imagens usadas para validação e atingiu a acurácia de 99%. A solução via Ensembles DDVR atingiu uma acurácia similar (98%) com apenas 680 imagens para treinamento. Os experimentos indicam que, enquanto o principal foco do DADA são as aplicações para análise de WSIs, mesmo aplicações de *deep learning* bem estabelecidas podem se beneficiar das soluções aqui apresentadas.

7.2 Resultados com a Aplicação de Análise de LITs

Essa seção analisa o aprendizado ativo com a tarefa de classificação de *patches* para a presença de LITs (Capítulo 5). Primeiramente, as estratégias de AL da literatura são avaliadas (Seção 7.2.1). Posteriormente, o foco é o impacto que o DADA possui nas mesmas condições (Seção 7.2.2) e seus efeitos na seleção de *patches* (Seção 7.2.3), bem como os efeitos de se utilizar *data augmentation* em conjunto com o DADA (Seção 7.2.4). Por fim, são conduzidos testes e análises sobre o tempo de execução das soluções discutidas (Seção 7.2.5).

Para o DADA e a aplicação motivadora de identificação de LITs o principal o modelo CNN utilizado é a InceptionV4 [183], o qual produz 1.536/4.608 dimensões de características nas configurações via MC Dropout/Ensembles. Outras redes são testadas para fins de avaliação da generalização das soluções, especialmente com relação à usabilidade do NAR. As redes são treinadas com o otimizador Adam e uma taxa de aprendizado inicial de 0.00005, reduzida a 1/10 após 30 épocas, usando tamanho de *batch* de 64 ou 96 imagens, a depender da rede utilizada ou do grau de simplificação aplicado.

O impacto dos parâmetros do DADA em seu desempenho é apresentado na Seção 7.3. Nos experimentos aqui desenvolvidos é utilizada a melhor configuração, composta por: PCA com vetor de características com 50 dimensões, 20 clusters e regeneração do *subpool* (α) a cada 2 iterações de aquisição. Em todos os casos, as redes são treinadas por completo, sem nenhum tipo de pré-inicialização de pesos de treinamentos prévios.

7.2.1 Desempenho dos Métodos de AL Existentes na Classificação de LITs

Nessa seção são avaliadas as abordagens CoreSet, Bayesiana e por Ensembles sem aplicação de nossa solução DADA. O tempo de execução de cada uma foi fixado em 40 horas (utilizando 2 GPUs NVIDIA V100 16GB GDDR), devido a limitações do tempo permitido pelo escalonador de serviços do ambiente computacional utilizado, ou a 3.500 *patches* adquiridos.

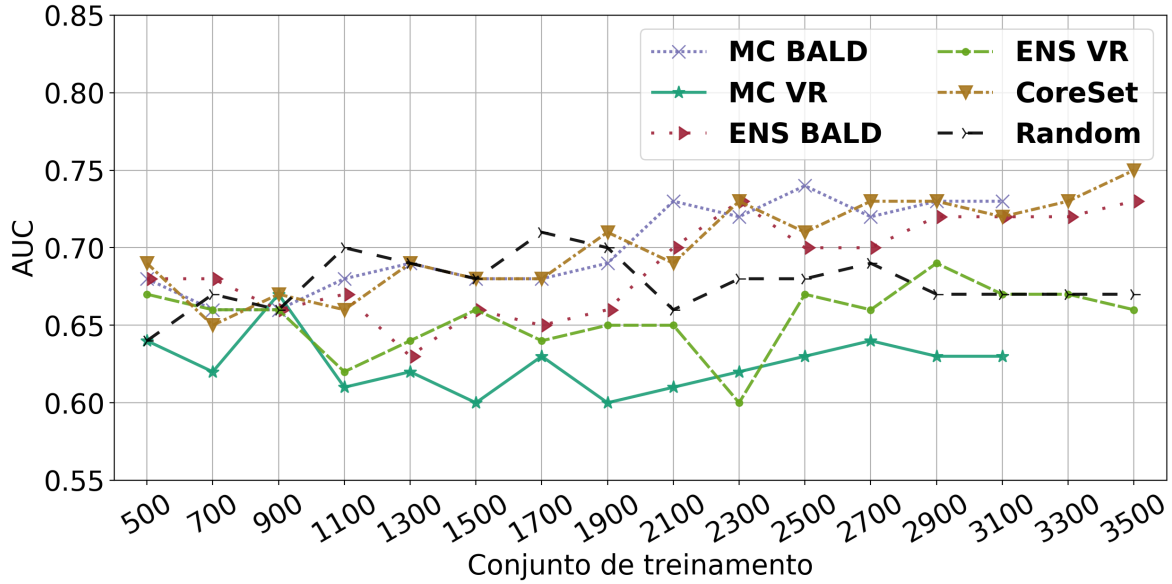


Figura 7.2: Estado da arte, com um *pool* estático para aquisições de 100.000 *patches*.

Os resultados são apresentados na Figura 7.2. A abordagem Bayesiana (MC) adquiriu apenas 3.100 patches em 40 horas, enquanto o método por Ensembles e CoreSet puderam obter os 3.500 *patches* estabelecidos. A métrica VR apresentou desempenho baixo tanto na estratégia por Ensembles quanto na Bayesiana, atingindo valores de AUC abaixo da aquisição aleatória. Vale notar que VR foi a melhor opção na base MNIST, demonstrando que a métrica de incerteza apresenta desempenhos variáveis, dependendo da tarefa em que é empregada. Esse comportamento, inclusive, pode ser considerado esperado para essa

configuração, visto que a base de LTIs é severamente desbalanceada enquanto a base MNIST mantém o balanceamento entre suas classes (referir à Seção 3.2.1).

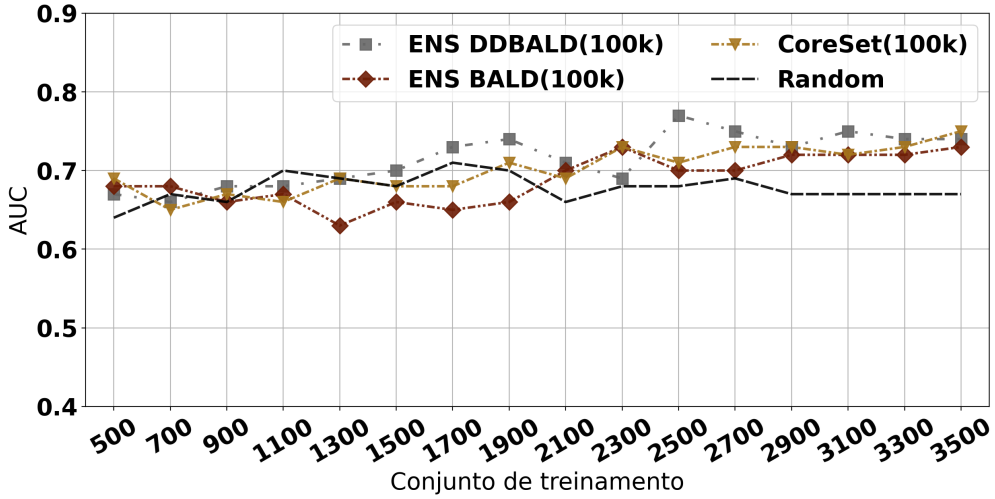
Tanto CoreSet, Bayesiano e Ensembles, utilizando a métrica BALD, produziram níveis de AUC comparáveis, se mantendo acima do nível da seleção aleatória apenas após a aquisição de 2.100 *patches*. Enquanto as abordagens por Ensembles e Bayesiana obtém níveis de AUC similares, Ensembles e o CoreSet são aproximadamente 2× e 4× mais rápidas que a opção bayesiana, nessa ordem. Dessa forma, nos experimentos seguintes, comparamos nossa solução com as abordagens Ensemble BALD e CoreSet apenas.

7.2.2 Impacto do DADA e *Subpools* no Desempenho de Classificação

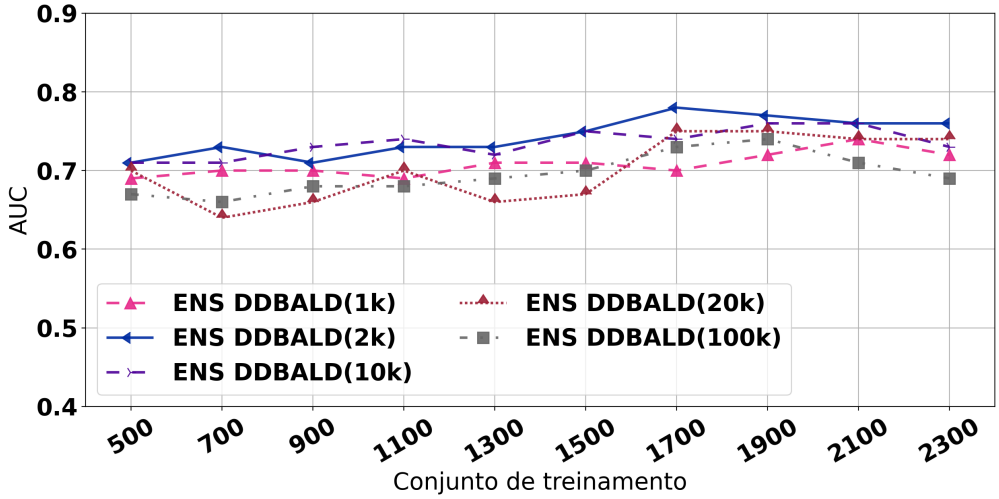
Os experimentos a seguir apresentam uma avaliação preliminar do impacto que o DADA possui sobre os níveis de AUC obtidos, bem como a influência que o tamanho dos *subpools* tem sobre essa métrica (Figura 7.3). Inicialmente, foram realizados testes que comparam o DADA às demais soluções avaliadas: Ensemble BALD, CoreSet e Random (Figura 7.3a). Em seguida, foram conduzidas 10 iterações de aquisição de *patches*, utilizando *subpools* que continham 2.000, 10.000 e 20.000 *patches* não rotulados, renovados a cada 2 aquisições, a fim de observar a possível influência que o tamanho dos *subpools* possa ter sobre a qualidade de classificação do modelo treinado.

Na Figura 7.3b, os tamanhos dos *subpools* são mostrados após o nome das estratégias (por exemplo, 2k correspondendo a 2.000) e o caso indicado como 100k indica a utilização da totalidade do estoque de imagens disponibilizadas. Para a avaliação da influência do tamanho do *subpool* foram realizados experimentos a partir da abordagem via Ensembles apenas, visto que para grandes conjuntos de dados não rotulados essa estratégia é significativamente mais rápida que a abordagem Bayesiana.

Os resultados experimentais apresentados na Figura 7.3a mostram que o Ensemble BALD utilizando DADA mas sem a aplicação do *subpool* dinâmico (ENS DDBALD (100k)) superou a versão do Ensemble BALD que não utiliza DADA (ENS BALD (100k)) e o CoreSet, que foram as melhores estratégias da Seção 7.2.1. Adicionalmente, a redução do tamanho do *pool* resultou em uma melhoria no resultado das previsões (Figura 7.3b). Como exemplo, ENS DDBALD (2k) atinge nível de AUC máximo de 0,78 utilizando 1.700 *patches* enquanto e o ENS DDBALD (100k) chega a AUC de 0,77 após 2.500 *patches* adquiridos. ENS BALD (100k), por outro lado, atinge AUC máximo de 0,73, enquanto o CoreSet se limita a 0,75. Esses resultados sugerem que o tamanho de 2.000 *patches* não rotulados seja o ideal para o *subpool* regenerado, visto que uma redução ainda maior, para um conjunto de 1.000 *patches* provocou piora do nível de AUC.



(a) Estratégias iniciais



(b) Subpools

Figura 7.3: Impacto do DADA e subpools.

Foi executada ainda uma comparação sistemática (e cara) sobre a versão gulosa do CoreSet e sobre a versão Ensembles BALD (os melhores métodos na literatura para nossa aplicação) em relação às melhores soluções aqui propostas: DADA com subpool de 2.000 *patches*, para as abordagens por Ensembles (ENS DDBALD) e MC Dropout (MC DDBALD). É importante salientar que a inclusão da versão Bayesiana nesse experimento foi possível graças à redução do tempo de processamento demandado com o uso de *subpool* dinâmico reduzido, o que não seria o caso para os experimentos apresentados anteriormente.

O experimento consistiu de 5 execuções de cada abordagem, para a produção da média e intervalo de confiança (IC) equivalente a um desvio padrão. Como demonstrado

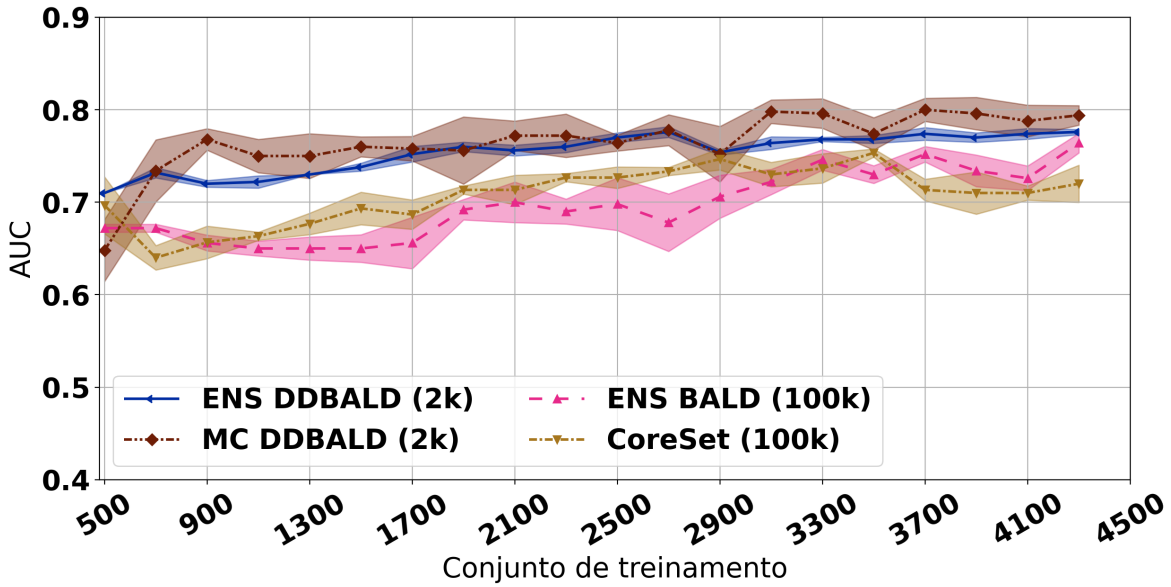


Figura 7.4: IC sobre as médias das principais soluções.

na Figura 7.4, o AUC médio das soluções que usam DADA apresentam crescimento em maior velocidade quando comparado às demais. Ainda, ENS DDBALD (2k) possui um intervalo de confiança mais restrito, o que demonstra a estabilidade do método em termos de velocidade de aprendizado. Finalmente, ENS DDBALD (2k) e MC DDBALD (2k) atingem um nível médio de AUC no decorrer das aquisições realizadas de 0,75 e 0,76, respectivamente, em contraste aos níveis de 0,70 obtidos pelo ENS BALD e 0,71 pelo CoreSet.

Após a aquisição de 4.000 patches as soluções atingem um *platô*. Essa situação pode ter como causa a inserção de *patches* informativos em quantidade insuficiente ou mesmo por limitações relativas à própria base de dados de treinamento. Para investigar essa situação, foi treinado um modelo de classificação sobre conjuntos de dados selecionados aleatoriamente a partir do total disponível, concatenados em uma série de experimentos devido às limitações de tempo de processamento para cada sessão de execução (Figura 7.5).

Em cada experimento foram adicionados 10.000 *patches* ao conjunto de treinamento, até que todos os 100.000 *patches* fossem usados. O nível de AUC máximo obtido foi de 0,76, com um conjunto de 60.000 *patches* para treinamento, sugerindo que exista um limite próximo a esse valor até onde a rede CNN utilizada pode atingir na classificação dessa base de dados. Possivelmente, o ruído inerente aos rótulos dados aos *patches* durante a formação do banco de dados traz dificuldades de aprendizado, que são mais perceptíveis quando um grande número de *patches* é usado.

É importante destacar que as abordagens ENS DDBALD (2k) e MC DDBALD (2k) atingiram AUC máximo de 0,78 utilizando apenas 2.700 *patches*, na média de experimen-

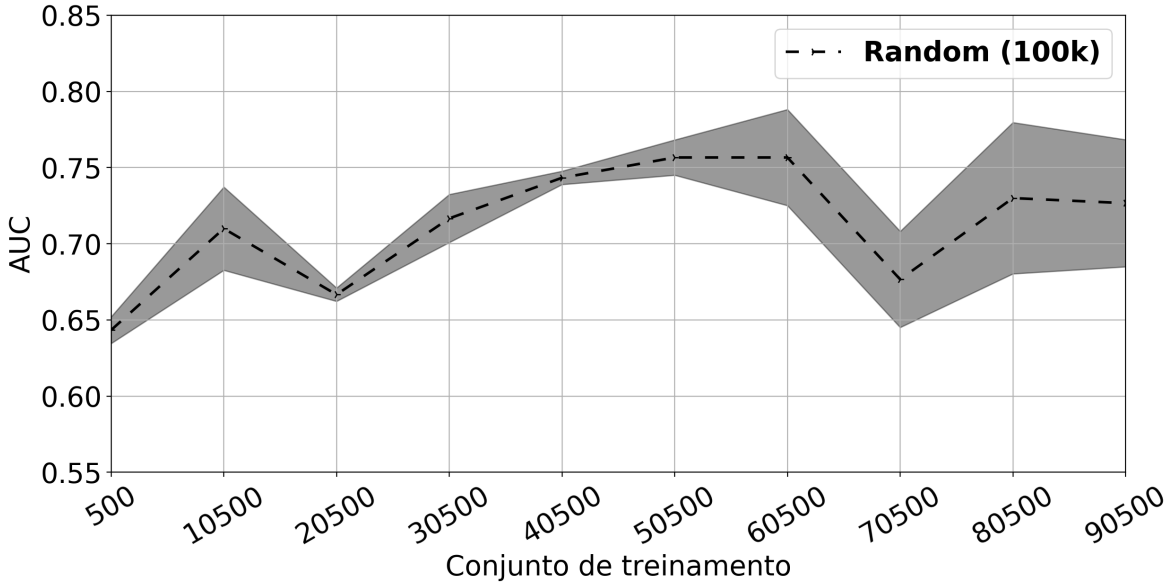


Figura 7.5: Aquisição aleatória de *patches* (até 90.500).

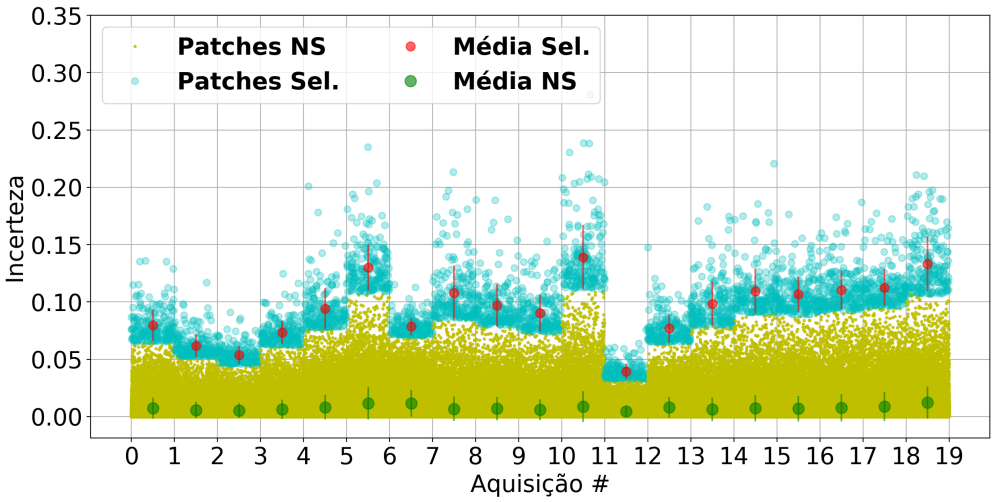
tos da Figura 7.4. Esses resultados demonstram que DADA rapidamente seleciona as imagens com maior potencial informativo e maximiza o nível de AUC com uma quantidade de pontos de treinamento muito reduzida. Assim, a partir dos dados observados nos experimentos até aqui conduzidos, adotar-se-á a definição **ENS DADA** e **MC DADA** para designar as versões por Ensembles e MC Dropout do DADA, utilizando *subpools* dinâmicos com 2.000 *patches* regenerados a cada 2 iterações e usando a função de incertezas BALD, exceto onde indicado o uso de configuração distinta.

7.2.3 Entendendo os Efeitos do DADA na Aquisição de Dados

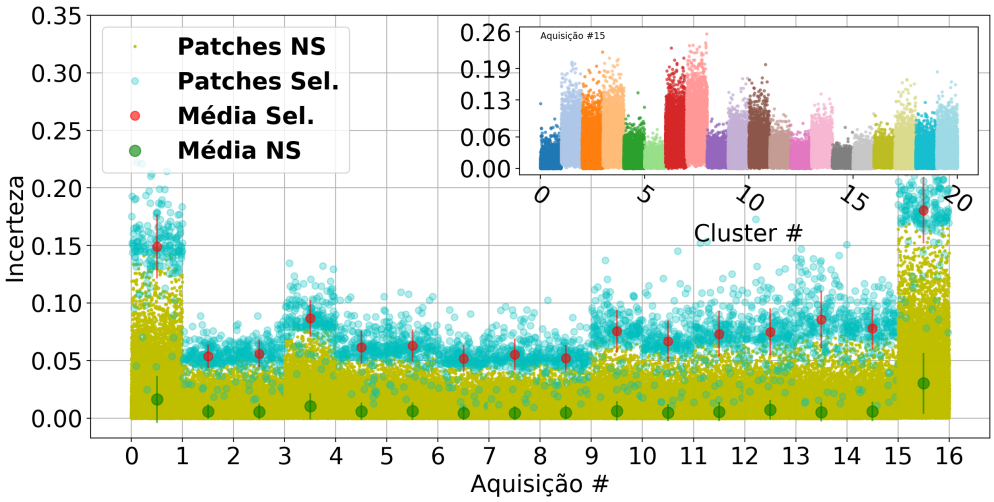
Os experimentos a seguir buscam entender como o DADA seleciona os *patches* do ponto de vista das características dessas imagens, visualmente e com relação à distribuição espacial nos *slides*, e avaliam o impacto dessa solução para os valores de incertezas apresentados pelos *patches*. A Figura 7.6 mostra os valores de incerteza dos *patches* selecionados pelas estratégias ENS BALD (100k) e ENS DADA (100k) à medida que as aquisições ocorrem. Enquanto o ENS BALD (100k) seleciona aqueles pontos com os maiores níveis de incertezas da totalidade do *pool*, DADA utiliza *clusters* para organizar os *patches* e seleciona aqueles de maior incerteza pertencentes a cada *cluster*.

As incertezas dos *patches* que compõem cada *cluster* em uma determinada etapa de aquisição (aquisição 15 de 20 do experimento) do ENS DADA (100k) são mostradas no quadro superior direito da Figura 7.6b. Observa-se que cada conjunto apresenta diferentes níveis de incerteza. Dessa forma, DADA permite que *patches* agrupados em um conjunto com nível de incerteza menor que os demais, mas que apresentam diferentes características

de tecido, sejam selecionados. Esse efeito também é demonstrado no painel de *patches* da Figura 6.2. Conclui-se, pela figura, que a maioria dos *patches* selecionados continua sendo daqueles de maior incerteza do *pool*, havendo, contudo, um compromisso entre incerteza e diversidade que beneficia as características visuais das imagens.



(a) ENS BALD(100k)



(b) ENS DADA (100k)

Figura 7.6: Incertezas de *patches* selecionados (Sel.) e não selecionados (NS) para as estratégias de *pool* fixo (100k).

As avaliações foram estendidas para verificar a distribuição espacial nos WSIs dos *patches* adquiridos. Para isso, eles foram agrupados utilizando o algoritmo K-Means, com base nas coordenadas (X,Y) que indicam a posição dos *patches* nos slides, sendo medidas as distâncias médias dos *patches* em relação aos centros dos clusters, para cada um deles, em todas as aquisições. Os resultados são mostrados na Figura 7.7.

Em todos os casos, a distância intra-cluster dos *patches* adquiridos em cada iteração aumenta, todavia, no caso do ENS BALD (100k) e DADA (100k), com *pool* fixo de tamanho 100.000, há uma aparente restrição a esse aumento, que muda de comportamento após cerca de 1.200 *patches* adquiridos. Isso sugere que inicialmente as soluções buscam trazer diversidade espacial, que é limitada pelo universo de *patches* disponíveis no *pool*. Interessante notar que o ENS DADA apresenta um comportamento crescente e aproximadamente linear, demonstrando que o uso de *subpools* não reduz o espaço de decisão em termos de localização, podendo, na verdade, aumentar a variabilidade espacial.

Por fim, o CoreSet seleciona *patches* que estão distantes uns dos outros no início, mas essa tendência se altera à medida que o conjunto de treinamento aumenta. Isso é uma consequência de sua definição acerca da seleção de pontos, que busca minimizar as distâncias máximas entre os pontos selecionados e não selecionados (Seção 4.1).

Desses experimentos se observa que além da diversidade visual introduzida pelo DADA, ele rapidamente selecione *patches* de áreas diferentes e incertas da imagem. Esse achado é um dos argumentos que explica o motivo pelo qual DADA melhora o aprendizado de uma CNN, permitindo que a rede seja treinada a partir de *patches* diversos, ao mesmo tempo, de alta incerteza e representativos.

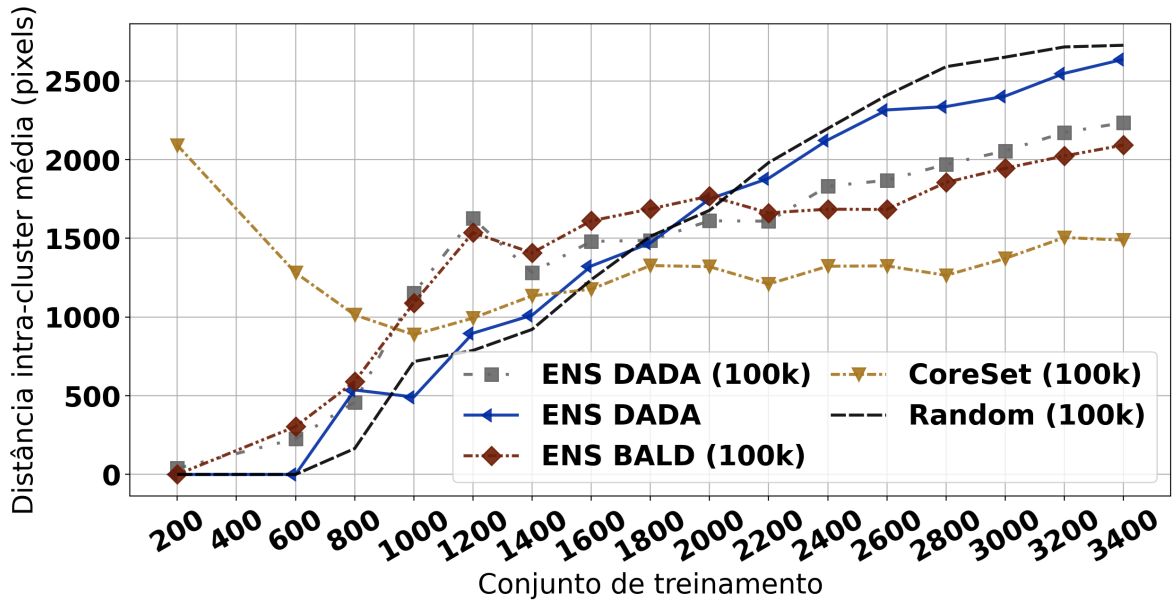
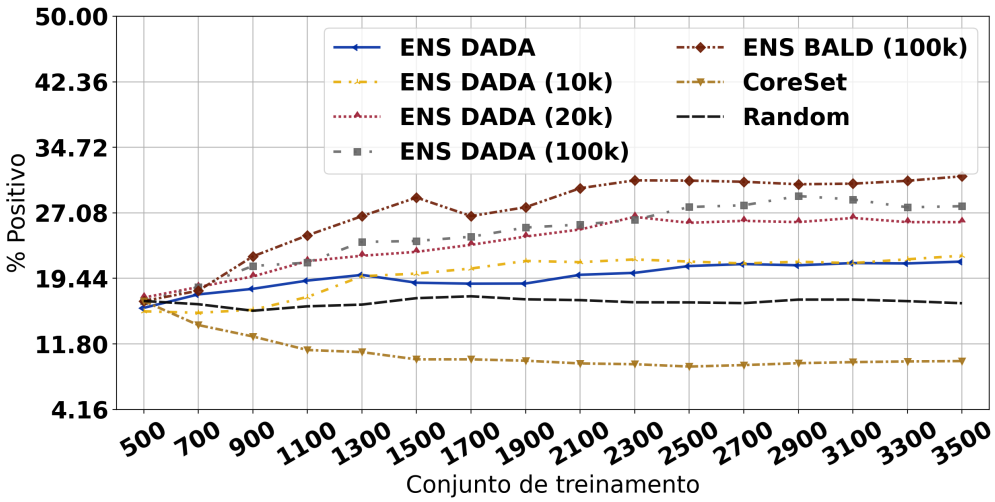


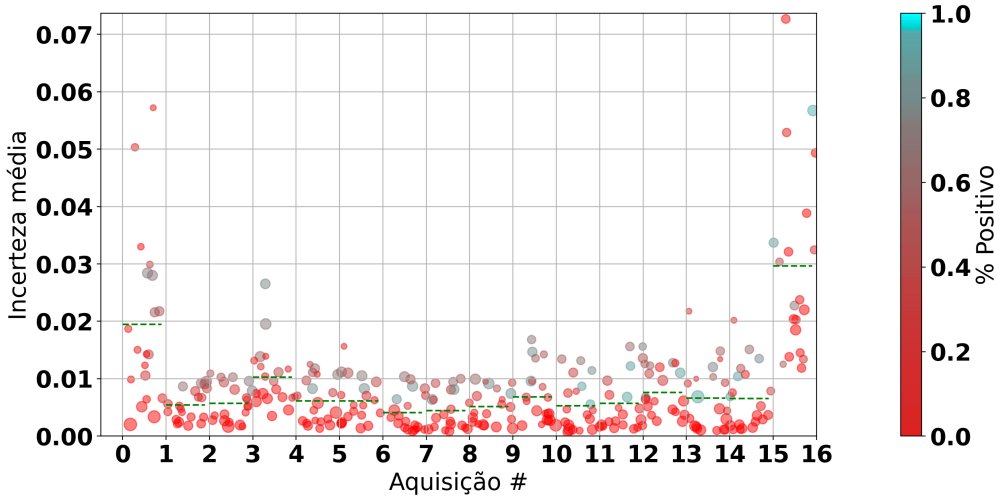
Figura 7.7: Distribuição espacial dos patches adquiridos ao redor dos centros dos clusters.

Foi constatado também que o rótulo dos *patches* adquiridos é impactado pelo DADA, como demonstrado na Figura 7.8. Imagens positivas, em geral, possuem valores de incertezas mais elevados, de forma que a abordagem por Ensembles BALD original captura um maior número de exemplos positivos. Isso pode ser visto na Figura 7.8a.

A Figura 7.8b mostra os percentuais de *patches* LIT-positivos/-negativos em cada cluster (cada ponto representa um cluster) produzido pelo DADA, na versão por Ensembles e estoque fixo de 100k imagens. Na figura é possível observar que determinados *clusters* possuem predominância de *patches* positivos (tonalidade azulada) enquanto o contrário acontece com outros (de tonalidade avermelhada). DADA limita a quantidade de exemplos positivos selecionados, dado que clusters com nível de incerteza menores também contribuem para a aquisição para garantir diversidade. Os *subpools* reduzidos adicionam outro limitador para a quantidade de *patches* positivos obtidos, tendo em vista a menor quantidade deles disponível no *pool*.



(a) % de *patches* LIT-positivos adquiridos.



(b) Composição dos *clusters* LIT-positivos/-negativos. Cada ponto corresponde a um *cluster* do ENS DADA (100k).

Figura 7.8: Classificação (pos/neg) de *patches* adquiridos pelas estratégias de AL.

7.2.4 Avaliando a Utilização de Data Augmentation (DA) no Processo de Aquisição

Data augmentation (ver Seção 2.2.2) foi utilizado em experimentos de aquisição do DADA com a aplicação de um conjunto de operações diretamente sobre o *batch* de imagens fornecidas à rede, durante o treinamento. Essas operações foram escolhidas a partir dos experimentos já conduzidos em [163], sendo bem sucedidas em melhorar o desempenho de classificadores no contexto da identificação de LITs. Optou-se por aplicar as operações em tempo de execução pelo fato que nem todas as operações do conjunto são executadas ao mesmo tempo sobre cada *batch*, sendo definido um arranjo de 3 operações do conjunto total que serão aplicadas às imagens, de forma a produzir uma maior variabilidade aos *patches*.

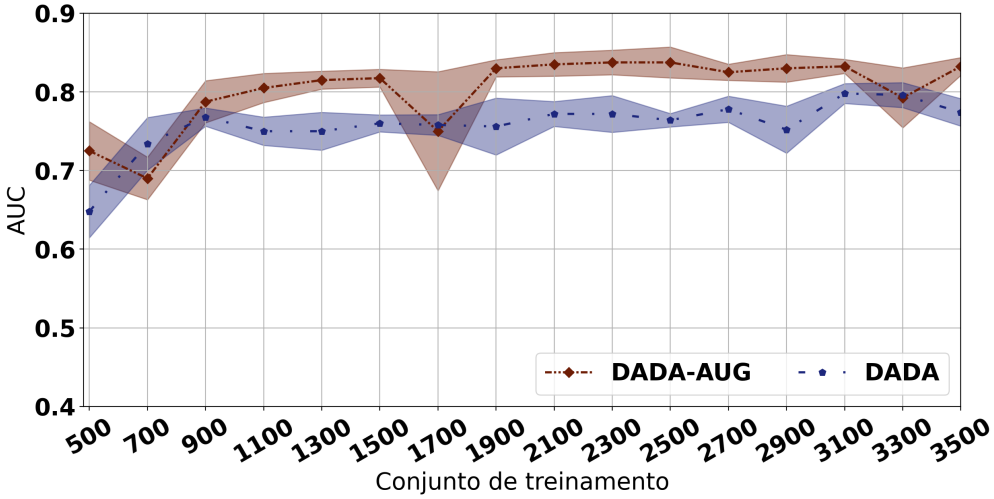
A aplicação dessas operações é realizada com o auxílio da biblioteca *imgaug* [186] para a linguagem Python. Com ela é possível definir o conjunto de operações desejadas e indicar uma seleção aleatória de um subconjunto delas para serem efetivamente aplicadas sobre um *batch* de imagens. Adicionalmente, a aplicação das operações durante a execução dos experimentos não produziu reflexos significativos no tempo de execução das iterações.

A Figura 7.9 traz os resultados dessa estratégia sobre o nível de AUC obtido para a rede InceptionV4, tanto para a abordagem probabilística do MC Dropout quanto com o uso de Ensembles. A Figura 7.9a retrata a abordagem do MC Dropout e percebe-se que há um ganho significativo de desempenho de classificação do DADA com o uso de DA em relação à versão original. Enquanto o DADA puro apresenta uma média de AUC de 0,76 considerando todas as iterações de aquisição sobre 5 repetições de experimentos, a aplicação de DA eleva esse nível para 0,80, um ganho 5,3%.

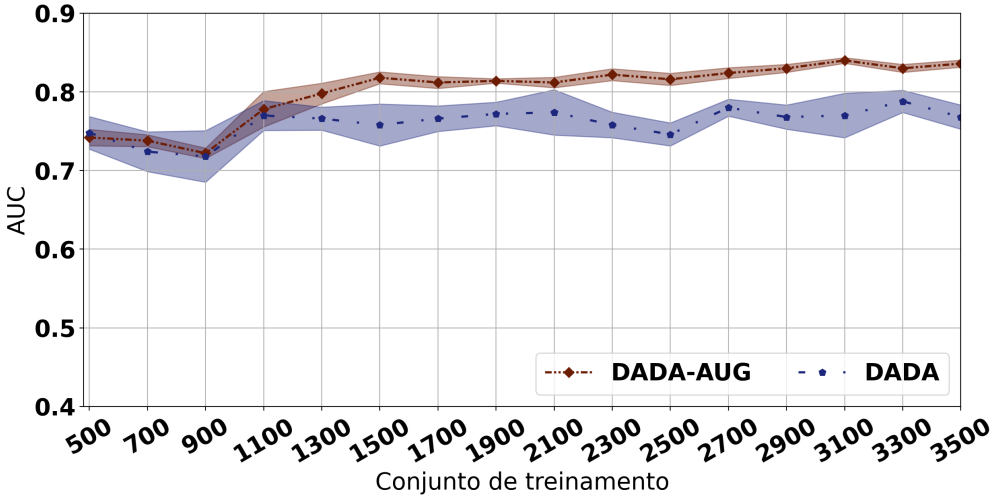
O mesmo comportamento é observado também para a abordagem via Ensembles, em que a versão com DA apresentou um nível de AUC médio de 0,80 enquanto o DADA puro obteve 0,76 (Figura 7.9b), com um ganho médio dos mesmos 5,3%. Assim, apesar de *data augmentation* não ser uma estratégia que visa criar um conjunto de treinamento, ela é uma importante ferramenta complementar ao DADA, permitindo que uma quantidade menor de *patches* seja necessária para se obter um nível de classificação em particular.

7.2.5 Tempo de Execução do Aprendizado Ativo

Aprendizado ativo é um processo computacionalmente caro para a análise de imagens de patologia. Dessa forma, enquanto o principal objetivo do aprendizado ativo é ter a capacidade de melhorar o nível de AUC do modelo usando o mínimo de imagens, reduzir o tempo de execução é importante. Nessa seção, são apresentados os tempos de execução das estratégias de AL previamente estudadas e o impacto que o uso de *subpools* possui. As



(a) **MC Dropout**: níveis de AUC com utilização de *data augmentation*.



(b) **Ensembles**: níveis de AUC com utilização de *data augmentation*.

Figura 7.9: *Data augmentation* (DADA-AUG) em comparação a DADA puro.

funções de incerteza BALD e VR têm tempos de execução muito próximos um do outro, considerando a estratégia Bayesiana ou por Ensembles, dessa forma o foco da discussão aqui desenvolvida é o BALD. Os experimentos não têm como foco realizar medidas de tempo rigorosas e foram executados em uma máquina compartilhada, usando 2 GPUs NVIDIA V100 com 16 GB de memória de vídeo dedicada e *batches* de 64 imagens.

A Figura 7.10 mostra o tempo de execução do AL para cada iteração, sendo dividido em tempo de treinamento (porção inferior das barras) e tempo de aquisição (porção superior das barras). A aquisição inclui a lógica para calcular incertezas e selecionar *patches* e o treinamento se refere ao processo de aprendizado da CNN após a inserção de novos *patches* no conjunto de treinamento.

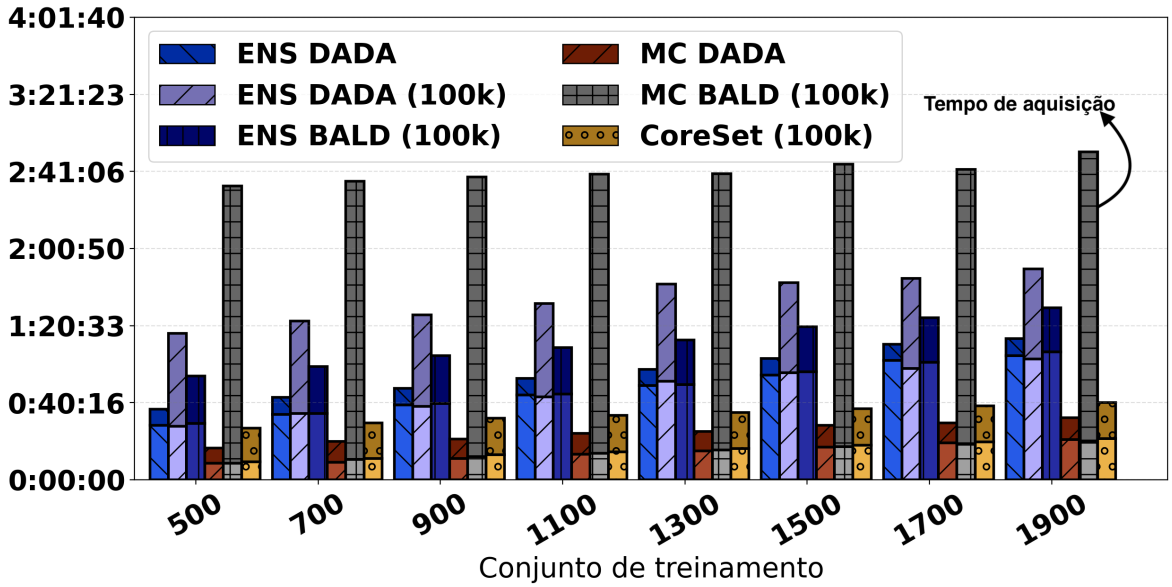


Figura 7.10: Tempo de execução do AL por etapa de aquisição (iteração).

A estratégia Bayesiana é significativamente mais lenta que a opção por Ensembles, o que é uma consequência do tempo de aquisição da primeira, o qual demanda múltiplas (20) passagens da rede pelos dados do *pool*, contra 3 passagens no caso dos Ensembles (uma por modelo). Considerando todas as iterações da Figura 7.10 e o mesmo tamanho do *pool* (100k), a versão por Ensembles (ENS BALD (100k)) é 1,92× mais rápida que a versão Bayesiana (MC BALD (100k)), enquanto o CoreSet é 4,01× mais rápido.

Além disso, o ENS DADA (100k) apresentou tempo de execução 1,75× maior que ENS BALD (100k), devido às etapas para cálculo de extração de *features*, PCA e formação de clusters, executadas pelo DADA. Todavia, o uso de *subpools* em conjunto com DADA (ENS DADA) melhora o desempenho em relação ao tempo de maneira considerável. Com essa modificação, ENS DADA é 1,21× e 2,12× mais rápido do que ENS BALD (100k) e ENS DADA (100k), respectivamente. O CoreSet é uma abordagem com baixo tempo de execução, sendo 1,8× mais rápida que ENS DADA, mas é consistentemente ultrapassado pelo último com respeito aos níveis de AUC.

O melhor tempo foi obtido pelo MC DADA, que foi expressivamente beneficiado com a aplicação dos *subpools* dinâmicos. Nessa configuração, o MC DADA foi 6,5× mais rápido, na média das iterações, que a versão com *pool* fixo de 100.000 *patches*. Mesmo com essa marcante melhoria, o tempo de espera médio entre as iterações é de 24min 45s, o que implica em um alto custo em tempo de mão de obra de um especialista realizando anotações. Assim, novos avanços são necessários, o que motivou o desenvolvimento do NAR, cujos resultados são discutidos nas próximas seções.

7.2.6 Impactos do Uso do NAR

O NAR (Seção 6.3.1) foi desenvolvido com o objetivo de reduzir o tempo total de cada iteração de aquisição de *patches* para anotação pelo DADA. A redução de tempo é um resultado esperado da redução da quantidade de operações matemáticas (de ponto flutuante nesse caso específico - FLOPs) necessárias tanto para treinar um modelo quanto para se realizar uma previsão. Como já discutido, a redução de FLOPs deverá ter uma dupla influência na redução do tempo de cada iteração, tendo em vista que ambas as etapas (treino x previsão) são afetadas.

A influência sobre essas duas etapas é diferente para cada abordagem de cálculo de incertezas usada visto que MC Dropout tem o tempo de iteração dominado pela fase de cálculo de incertezas (múltiplas passagens sobre os dados não rotulados), enquanto no caso dos Ensembles o treinamento é a etapa dominante, pela necessidade de se treinar múltiplas redes.

As subseções a seguir abordam a aplicação do NAR sobre três ângulos: redução de FLOPs dos modelos simplificados em relação a um nível de AUC, reflexo sobre o tempo de cada iteração do DADA e sobre a qualidade da seleção de *patches*.

Número de FLOPs das Redes

O número de FLOPs consumidos por uma rede é uma característica do modelo, considerando um tamanho de imagem de entrada e a quantidade de classes que o modelo deve reconhecer. Além de trazer os resultados obtidos com o NAR, a Tabela 7.8 também apresenta os valores obtidos pelo emprego da técnica ResRep (Seção 4.2) sobre a rede ResNet50 V2, para fins de comparação. As características dos modelos gerados foram trazidas pelas Tabelas 4.1 no caso do ResRep e 6.1 para o NAR.

Os experimentos conduzidos abaixo correspondem a execuções de treinamentos simples, em que os modelos (originais e reduzidos) são treinados sobre um conjunto de 3.800 imagens pré-selecionadas pelo DADA, mas não executam aprendizado ativo. Os valores de AUC são calculados após 50 épocas de treinamento, sendo aplicado *data augmentation* conforme exposto na Seção 7.2.4. Os valores contidos nas Tabelas 7.8 e 7.9 são médias de 3 repetições.

Da Tabela 7.8 se depreende que a rede original, para o tamanho de entrada definido e classificação binária, produziu um AUC máximo de 0,87, consumindo 9,65 GFLOPs. As redes reduzidas produzidas pelo ResRep foram geradas a partir de valores limiares de ϵ entre 0,82 e 0,94, sendo os maiores valores correspondentes a graus mais elevados de contração, para os quais se espera menor contagem de FLOPs. Assim como no trabalho original com a base de dados ImageNet, o ResRep também foi capaz de produzir redes

CNN	AUC	GFLOPs	Entrada	Camadas c/ Param.	# de Camadas
ResNet50 V2 [142]	0,87 +/- 0,01	9,65	240 × 240	50	225
ResNet ResRep $\epsilon = 0,82$	0,87 +/- 0,01	8,34			
ResNet ResRep $\epsilon = 0,84$	0,82 +/- 0,02	7,91			
ResNet ResRep $\epsilon = 0,86$	0,84 +/- 0,03	7,63			
ResNet ResRep $\epsilon = 0,88$	0,81+/- 0,05	7,68			
ResNet ResRep $\epsilon = 0,90$	**0,86+/- 0,05**	**7,27**			
ResNet ResRep $\epsilon = 0,92$	0,69+/- 0,03	6,10			
ResNet ResRep $\epsilon = 0,94$	0,73+/- 0,01	6,09			
ResNet NAR $\phi = 1$	0,84+/- 0,02	5.15	209×209	42	170
ResNet NAR $\phi = 2$	**0,86+/- 0,02**	**2,96**	181×181	36	160
ResNet NAR $\phi = 3$	0,80+/- 0,02	1,53	157× 157	30	134

Tabela 7.8: AUC, Giga-FLOPs (GFLOPs) correspondentes ao tamanho da entrada do modelo, número de camadas com pesos e total de camadas do modelo, para a ResNet50 V2 e as versões simplificadas pelo ResRep e NAR. Os intervalos de confiança correspondem a um desvio padrão da média.

simplificadas que mantém o desempenho de classificação sobre a base de LITs aqui desenvolvida. Conforme a tabela, o melhor resultado obtido com o ResRep ocorreu com $\epsilon = 0,90$, com um nível de AUC de 0,86 (equivalente ao original dentro do IC) e um total de 7,27 GFLOPs, correspondendo a uma redução de 24,7%.

O NAR, por sua vez, também obteve um nível de AUC equivalente ao original, considerando o IC para $\phi = 2$, contudo obtendo uma redução do número de FLOPs de 9,65 para 2,96 GFLOPs, uma redução de 70%, equivalente a $\frac{1}{3,26}$ do original. Esse valor se mostra próximo do esperado, conforme a formulação do NAR, que prevê uma redução do número de FLOPs proporcional a $1/2^{\phi}$ ($\frac{1}{4}$ no caso concreto).

CNN	AUC	GFLOPs	Entrada	Camadas c/ Param.	# de Camadas
Inception V4 [183]	0,920+/- 0,002	15,48	240 × 240	245	861
Inception IR 1	**0,910+/- 0,010**	**9,80**	209 × 209	245	861
Inception IR 2	0,890+/- 0,014	6,64	181 × 181		
Inception IR 3	0,880+/- 0,020	4,79	158 × 158		
Inception IR 4	0,870+/- 0,008	2,76	137 × 137		
Inception IR 5	0,860+/- 0,006	1,92	119 × 119		
Inception IR 6	0,770+/- 0,090	1,14	104 × 104		
Inception NAR $\phi = 1$	**0,910+/- 0,003**	**8,14**	209 × 209	206	723
Inception NAR $\phi = 2$	**0,920+/- 0,003**	**4,17**	181 × 181	179	627
Inception NAR $\phi = 3$	0,900+/- 0,004	2,21	158 × 158	145	507
Inception NAR $\phi = 4$	0,880+/- 0,005	1,02	137 × 137	123	429
Inception NAR $\phi = 5$	0,870+/- 0,006	0,56	119 × 119	101	351
Inception NAR $\phi = 6$	0,840+/- 0,001	0,28	104 × 104	91	315

Tabela 7.9: AUC, Giga-FLOPs (GFLOPs) correspondent to input sizes, number of parameter layers and total layers of Inception V4 and simplified networks produced by NAR.

A rede Inception V4 foi também analisada com relação aos níveis de AUC e FLOPs obtidos, a partir de 6 níveis de redução com o NAR. Ao contrário do que foi realizado para a ResNet50 V2, não foi possível aplicar o ResRep sobre a rede Inception (apesar de esforços para tal), tendo em vista a alta dificuldade de adaptação do método para essa

rede, que é consideravelmente mais profunda e de estrutura mais complexa. Todavia, para demonstrar a eficácia do NAR, foram realizados experimentos que mostram que a mera redução do tamanho de entrada da rede, apesar de contribuir para a redução do número de FLOPs, não possui desempenho tão bom quanto o NAR. A essa estratégia foi dado o nome de *Input Reduction (IR)*, o qual será utilizado doravante no trabalho.

A Tabela 7.9 apresenta os resultados obtidos. A partir dela, observa-se que a Inception V4, nas configurações estabelecidas, produziu AUC de 0,92, demandando 15,48 GFLOPs. As redes geradas pelo NAR obtiveram resultados muito próximos aos originais, para valores de $\phi = 1$ e $\phi = 2$, obtendo, respectivamente, reduções de 47,5% e 73,1% no número de FLOPs com relação à rede original. Esses valores correspondem a $\frac{1}{1,9}$ e $\frac{1}{3,71}$ da rede alvo, também em linha com a expectativa de redução de FLOPs para o NAR.

A estratégia de IR não produz qualquer alteração na estrutura da rede, exceto as dimensões de seus tensores, mas, como se observa da Tabela 7.9, é capaz de trazer ganhos de processamento que, mesmo associados a pequenas perdas de desempenho de classificação, podem ser aceitáveis em algumas aplicações. Mesmo assim, o NAR se mostra mais eficiente ao conduzir reduções coordenadas entre as três dimensões da rede. Ao se comparar um mesmo nível de AUC, por exemplo, 0,87, obtido pelo IR 4 e NAR $\phi = 5$, o total de FLOPs da solução IR 4 é 4,92× maior que o da rede gerada pelo NAR.

É interessante notar que a tendência esperada, de redução da capacidade de classificação conforme os fatores de simplificação aumentam, existe, dentro de intervalos de variações, de forma que modelos menores podem ainda apresentar resultados ligeiramente melhores que outros maiores, como ocorreu com a rede ResNet50 V2, tanto usando o NAR quanto ResRep. Adicionalmente, no ResRep, o mecanismo de escolha dos filtros a serem removidos depende dos gradientes obtidos durante o treinamento da rede, de forma que nem todo incremento do coeficiente ϵ gera uma redução do número de FLOPs, como se observa na Tabela 7.8, entre os valores $\epsilon = 0,86$ e $\epsilon = 0,88$.

Diante do exposto, é possível concluir que o NAR consegue produzir versões simplificadas de redes complexas que ainda possuem um desempenho de classificação elevado, mesmo com expressiva redução do número de operações de ponto flutuante executadas pelos modelos. Foi possível constatar também que as redes produzidas pelo NAR possuem desempenho superior àquelas produzidas pela técnica ResRep, técnica essa que se encontra no estado da arte dentro da área de simplificação de CNNs.

Sobre o Tempo de Execução das Iterações

Na seção anterior foram avaliadas técnicas de simplificação em seu estado puro, demonstrando que o NAR possui grande capacidade de redução de FLOPs e resistência à perda de capacidade de classificação. Nos experimentos conduzidos aqui, o NAR é avaliado

integrado ao DADA, buscando verificar como os tempos de cada iteração são afetados pelo seu uso. Salienta-se que esse é o maior objetivo de se utilizar redes mais simples no processo do aprendizado ativo, ou seja, reduzir o tempo necessário para a seleção de novas imagens a serem anotadas e, por consequência, diminuir o custo de anotação como um todo.

Os experimentos iniciais do DADA foram executados em uma máquina diferente daquela em que o NAR foi avaliado. As principais diferenças são com relação à GPU e ao sistema de arquivos, que tiveram significativas melhorias de desempenho. Os experimentos executados aqui se deram com o uso de uma GPU NVidia V100 com 32 GB de memória de vídeo dedicada, permitindo o uso de *batches* de 64 imagens que podem ser carregados em uma única GPU. Esses fatores alteram significativamente os tempos de execução, os quais são reduzidos quando comparados com os valores obtidos originalmente e mostrados na Figura 7.10. Dessa forma, os experimentos sobre o DADA original foram repetidos sob as novas condições. Adicionalmente, os valores plotados nas figuras a seguir são médias de 5 repetições de cada experimento.

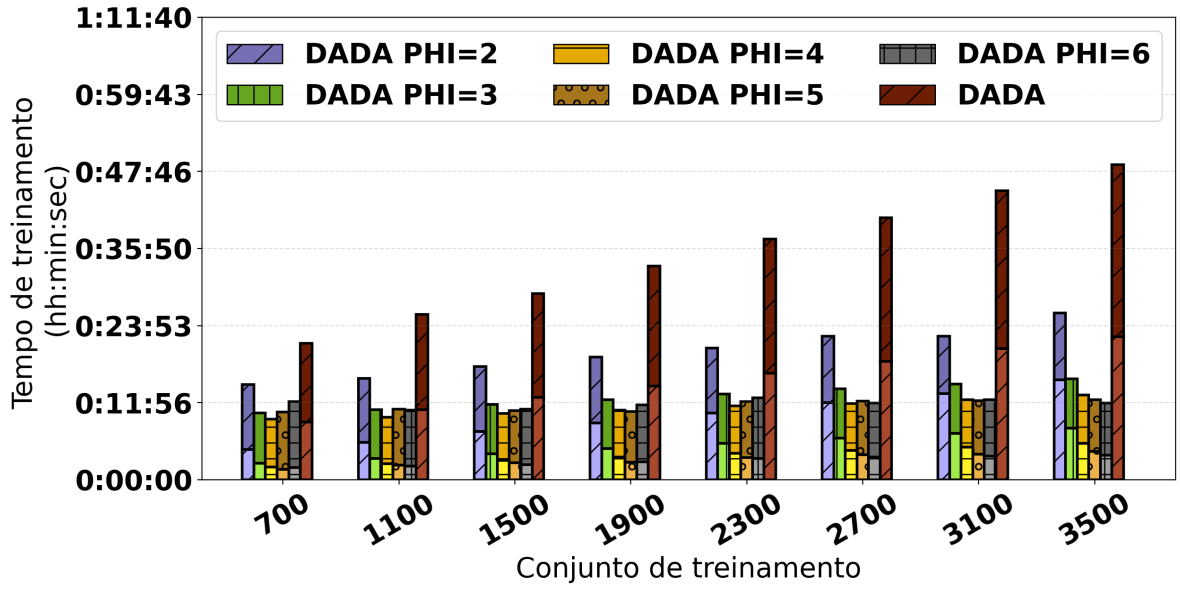


Figura 7.11: Tempo de execução do AL por etapa de aquisição (iteração), na abordagem por **MC Dropout**.

A Figura 7.11 ilustra os tempos de algumas das iterações de anotações executadas com a abordagem MC Dropout. As porções inferiores das barras correspondem ao tempo de treinamento enquanto as porções superiores são os tempos de aquisição de *patches* para anotação. As reduções nos tempos das iterações são expressivas e, conforme o tamanho do conjunto de treinamento aumenta, as diferenças de tempo também seguem essa tendência. Para a iteração de 700 *patches* o uso das redes reduzidas com $\phi = 2$ e $\phi = 3$ apresentam

uma redução respectivamente de 30,4% e 51,0% do tempo total de iteração sobre o DADA original, enquanto que para o conjunto de treinamento de 3.500 *patches* essa redução é de respectivamente 47,2% e 68,0%. Considerando os tempos médios de todas as iterações, as redes reduzidas $\phi = 2$ e $\phi = 3$ apresentam **reduções de 41,1% e 61,5%, respectivamente**.

As reduções efetivas de tempo de execução não correspondem exatamente à redução esperada para o número de FLOPs produzida pelo NAR. Os tempos obtidos equivalem a $\frac{1}{1{,}70}$ e $\frac{1}{2{,}60}$ do original, respectivamente, nos casos $\phi = 2$ e $\phi = 3$. Esse comportamento é esperado visto que o número de FLOPs é uma das variáveis envolvidas no tempo de execução, que também é afetado pelos tempos de leitura do sistema de arquivos, estado de carga da máquina, características dos processos em execução, entre inúmeras outras circunstâncias. Adicionalmente, as diferenças de tempo médio entre as versões $\phi = 4$ e $\phi = 6$ são pequenas, sendo 667,13 segundos a média no primeiro caso e 707,25 no segundo, um incremento inesperado de 6,01%. Essa constatação demonstra que outros gargalos passam a ter maior relevância que o tempo específico de processamento durante o treinamento/previsão das redes.

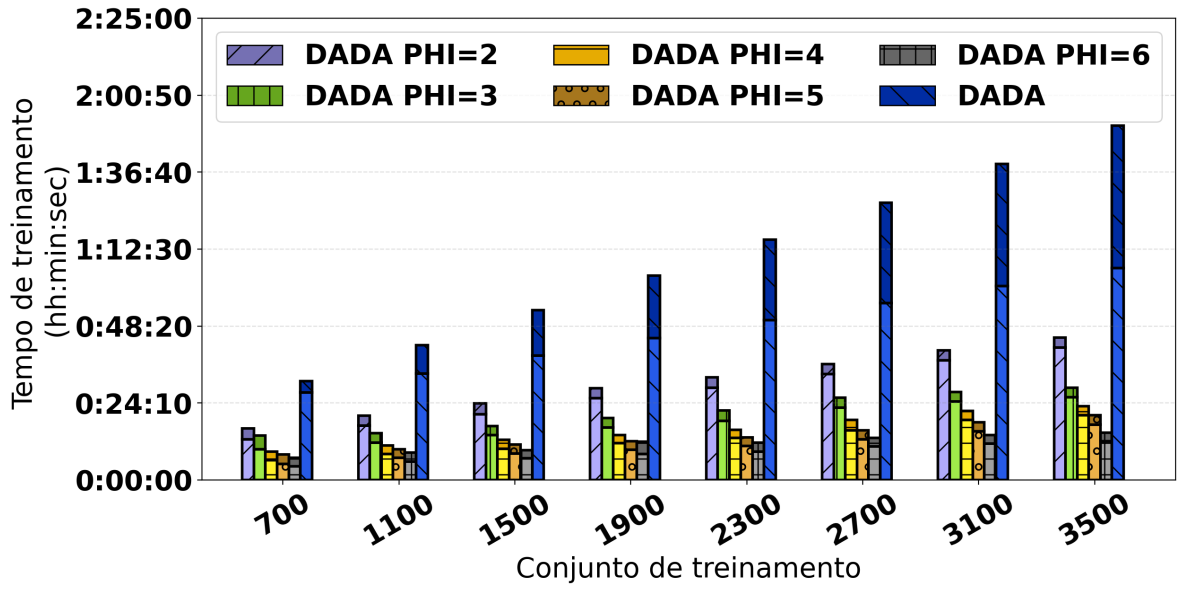


Figura 7.12: Tempo de execução do AL por etapa de aquisição (iteração), na abordagem por **Ensembles**.

Mesmo já tendo observado que a aplicação de *subpools* junto ao DADA faz com que a abordagem probabilística do MC Dropout torne essa vertente mais vantajosa em termos de tempos de iteração que o uso de Ensembles, foram realizados experimentos com essa modalidade de cálculo de incertezas em conjunto com o NAR para avaliar se a simplificação de redes produziria reduções de tempo de treinamento que voltassem a beneficiar

a abordagem por Ensembles. Os resultados são mostrados na Figura 7.12. Conforme se depreende da figura, os tempos de iteração utilizando Ensembles ainda se mantém mais elevados do que aqueles obtidos pela abordagem MC Dropout.

Apesar disso, o NAR foi capaz de produzir reduções expressivas nessa abordagem também. Diferente do que ocorreu com a abordagem Bayesiana, com Ensembles foram observadas reduções de tempo de execução consistentes até a versão DADA $\phi = 6$. As reduções de tempo com relação ao DADA, obtidas entre o conjunto com 700 *patches* e 3.500 *patches* foram, respectivamente, de 47,8% e 59,8% para a versão $\phi = 2$ e de 74,4% e 84,4% para a versão $\phi = 6$, sendo, nesse último caso, 6,41 $\times$ mais rápido que o DADA.

Os experimentos de tempo trazem uma forte indicação de que o NAR, junto ao DADA é de fato capaz de reduzir drasticamente os tempos necessários para se obter um novo conjunto de *patches* para anotação a cada iteração. Resta agora analisar se as imagens selecionadas de fato são capazes de treinar a rede alvo até níveis desejáveis de AUC, o que é feito a seguir.

Sobre a Qualidade de Seleção

Os *patches* selecionados pelo DADA com o uso do NAR devem ser usados para treinar a CNN alvo e fazer com que ela atinja os níveis de AUC esperados, ou seja, que sejam o mais próximo possível daqueles obtidos quando a própria rede alvo realiza a seleção de *patches*. As figuras a seguir mostram os resultados obtidos com essa configuração, utilizando-se a rede InceptionV4, inicialmente sem o uso de *data augmentation* e, posteriormente, com a aplicação da técnica. Será avaliado também se existe uma diferença de níveis de AUC que justifiquem o uso da abordagem por Ensembles, visto que seu custo computacional continua maior que a opção Bayesiana.

A Figura 7.13 compila os resultados obtidos com o uso de Ensembles e o NAR para 5 níveis de redução da CNN ($\phi = 2$ a $\phi = 6$) sem o uso de *data augmentation*. Considerando a média das iterações realizadas, o DADA original atinge AUC de 0,76, enquanto o nível 2 obtém AUC médio de 0,78, mesmo sendo 2,31 $\times$ mais rápido. Os níveis $\phi = 3$ e 4 atingem os mesmos 0,76 de média de AUC do DADA original, com reduções de tempo de iteração de 70% e 77,6%, se mostrando opções com a melhor relação AUC/custo. Como esperado, com reduções mais agressivas nas redes intermediárias, os níveis $\phi = 5$ e $\phi = 6$ demonstram perda de desempenho de classificação, com AUC médio, respectivamente, de 0,70 e 0,74.

Com o uso da abordagem MC Dropout, os resultados obtidos foram semelhantes àqueles apresentados pela versão com Ensembles. A Figura 7.14 ilustra as curvas de AUC para os mesmos 5 níveis de simplificação da rede alvo pelo NAR. O DADA original manteve uma média de AUC de 0,76. Assim como na versão por Ensembles, com o MC Dropout

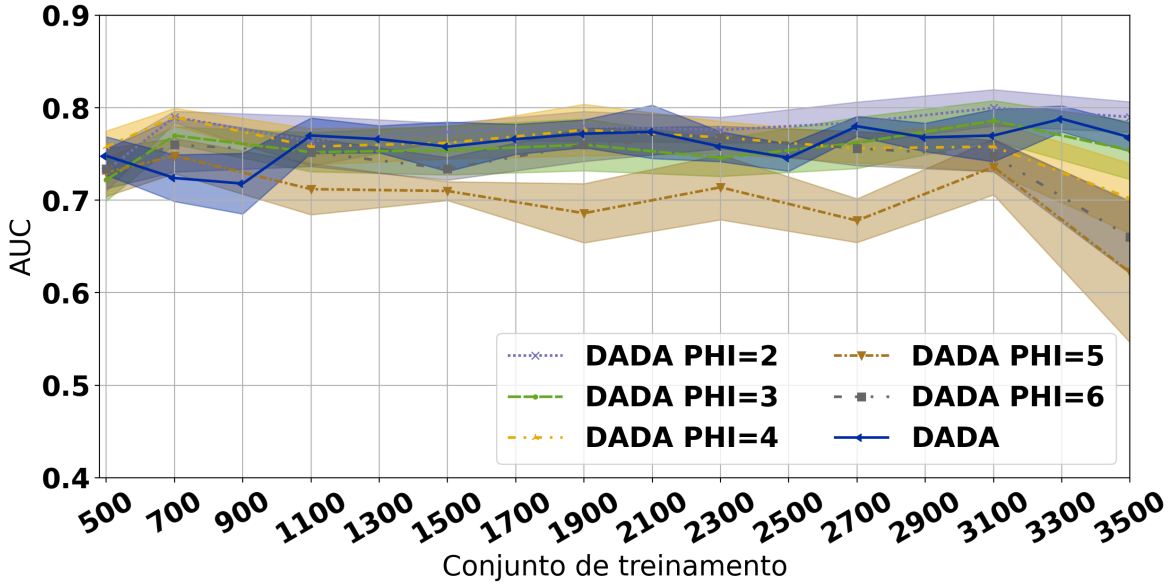


Figura 7.13: Níveis de AUC obtidos pela rede InceptionV4, com seleção de *patches* realizada pelas redes reduzidas pelo NAR, na abordagem por **Ensembles**.

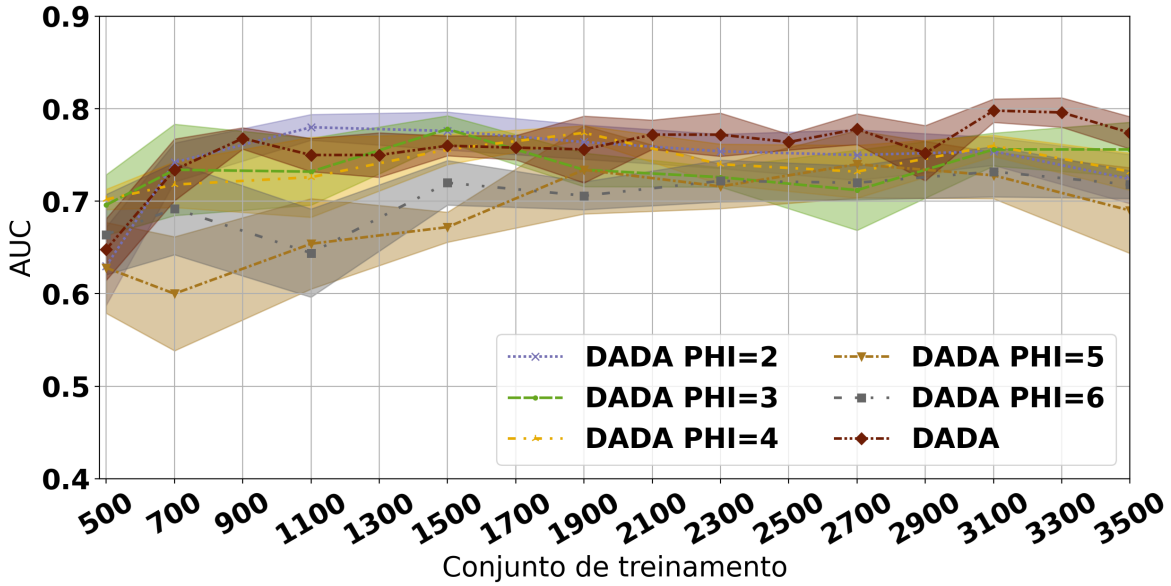


Figura 7.14: Níveis de AUC obtidos pela rede InceptionV4, com seleção de *patches* realizada pelas redes reduzidas pelo NAR, na abordagem **MC Dropout.**

os níveis de redução $\phi = 2$ a $\phi = 4$ obtiveram resultados satisfatórios, com médias de 0,74 para os 3 níveis, enquanto as reduções $\phi = 5$ e $\phi = 6$ tiveram perdas de desempenho, obtendo AUCs médios, respectivamente de 0,69 e 0,70. A consistência dos comportamentos observados nesses dois conjuntos de experimentos traz argumentos que indicam a veracidade da hipótese inicial, de que as redes mais próximas da rede alvo

poderiam produzir seleções de *patches* de boa qualidade.

O efeito do uso de *data augmentation* juntamente com o NAR foi avaliado e as Figuras 7.15 e 7.16 apresentam os resultados de AUC para os mesmos níveis de redução das redes. A aplicação de DA gerou uma elevação dos níveis de AUC da versão original do DADA, como já demonstrado na Seção 7.2.4. Essa elevação também ocorreu com os níveis obtidos a partir do uso das redes simplificadas pelo NAR.

Na abordagem por Ensembles (Figura 7.15), todas as versões reduzidas apresentaram resultados próximos uns dos outros, sendo que o DADA apresentou pico de AUC de 0,84 com 3.100 *patches* no conjunto de treinamento e média geral de 0,80. O nível $\phi = 2$ obteve pico de AUC de 0,83 com conjunto de treinamento de 3.500 *patches* e média de 0,79. Mesmo em uma redução mais drástica, como $\phi = 5$, o pico de AUC de 0,84 também foi obtido, em 3.500 *patches* e com média de 0,79.

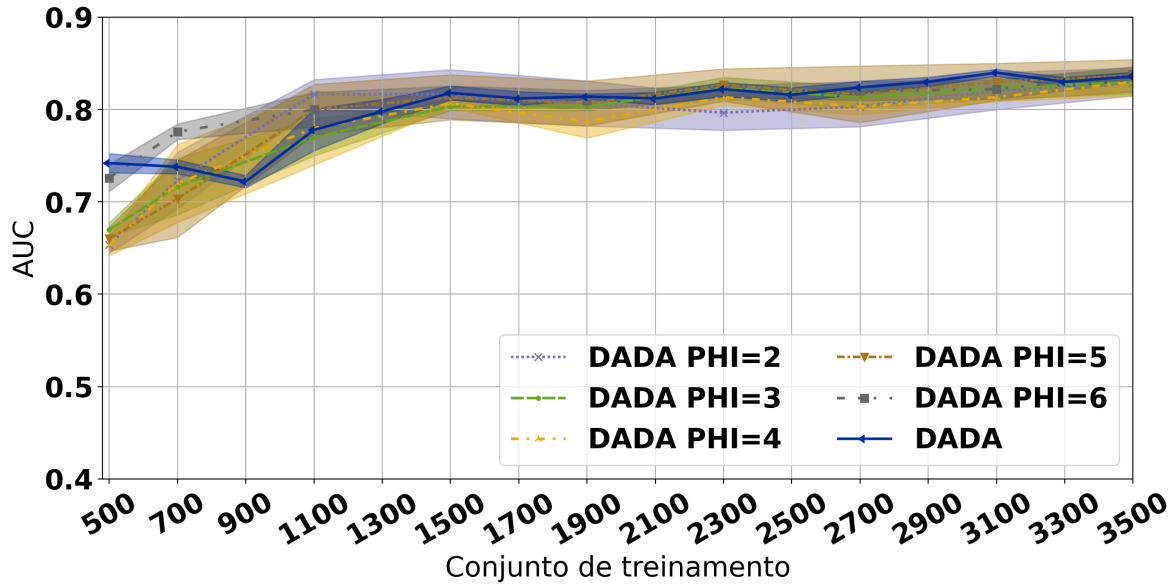


Figura 7.15: Níveis de AUC obtidos pela rede InceptionV4, com seleção de *patches* realizada pelas redes reduzidas pelo NAR e com a aplicação de DA, na abordagem por **Ensembles**.

A Figura 7.16 mostra as curvas de AUC para o DADA/NAR com o uso de *data augmentation* para a abordagem MC Dropout. O nível $\phi = 2$, com DA, foi capaz de prover níveis médios de AUC de 0,82. Conforme esperado, o nível $\phi = 6$ foi aquele que obteve a menor média, com AUC de 0,79, todavia próximo dos demais níveis e, inclusive, próximo dos valores do DADA. Se destacam os níveis 2 a 4, que produziram médias de AUC superiores ao DADA original, com valores de 0,82 nos três casos, mesmo com reduções no tempo de iteração de até 41,1% no caso do nível $\phi = 3$.

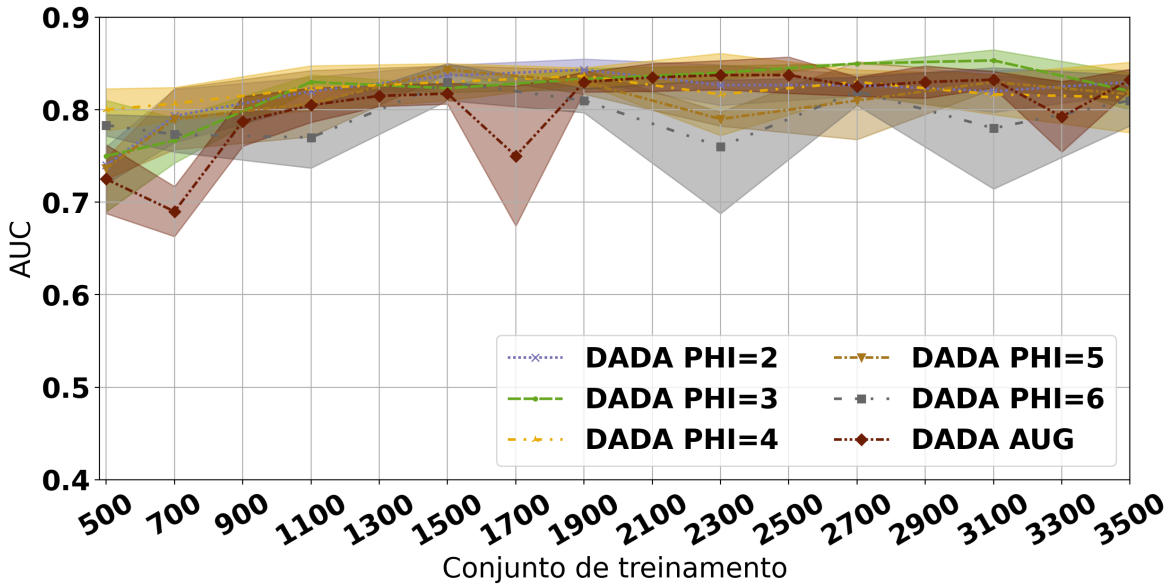


Figura 7.16: Níveis de AUC obtidos pela rede InceptionV4, com seleção de *patches* realizada pelas redes reduzidas pelo NAR e com a aplicação de DA, na abordagem **MC Dropout.**

Abordagem	**Original**						**Data augmentation**					
	DADA	$\phi = 2$	$\phi = 3$	$\phi = 4$	$\phi = 5$	$\phi = 6$	DADA	$\phi = 2$	$\phi = 3$	$\phi = 4$	$\phi = 5$	$\phi = 6$
MC Dropout	**0,76**	0,74	0,74	**0,74**	0,69	0,70	**0,80**	0,82	0,82	**0,82**	0,81	0,79
Ensembles	**0,76**	0,78	0,76	**0,76**	0,70	0,74	**0,80**	0,79	0,78	0,78	**0,79**	0,80

Tabela 7.10: Níveis de AUC médios obtidos com o uso do NAR com a rede InceptionV4.

O uso de DA com o NAR teve um efeito geral de aproximar os níveis de desempenho de todos os cenários, deixando os resultados das redes reduzidas mais próximos entre si e mais próximos do DADA. A Tabela 7.10 sintetiza os valores de AUC médios obtidos para todos os cenários avaliados sobre a rede InceptionV4. Dessa tabela é possível perceber tanto a mudança dos níveis médios de AUC já visualizados nas figuras mas também uma inversão entre qual a melhor abordagem (Ensembles ou MC Dropout) quando se aplica ou não *data augmentation*. De maneira geral, as médias da abordagem por Ensembles são maiores que aquelas obtidas pelo MC Dropout quando não se aplica DA, tendência que se inverte ao ser usada essa técnica. Em parte, essa observação pode ser explicada pela maior capacidade dos Ensembles em capturar as características das imagens fornecidas para treinamento (ver Seção 4.1.2), o que pode ser compensado pela variabilidade adicionada pelo DA.

O NAR foi avaliado também sobre outras arquiteturas de CNNs já bem estabelecidas, como a ResNet50 V2 [142] e a Xception [187], sem o uso de *data augmentation* e pela abordagem MC Dropout. O objetivo foi verificar se o comportamento observado com a Inception V4 também seria observado com outras redes. A Figura 7.17 apresenta os resultados da rede ResNet50 V2, com comportamento semelhante ao observado nos demais

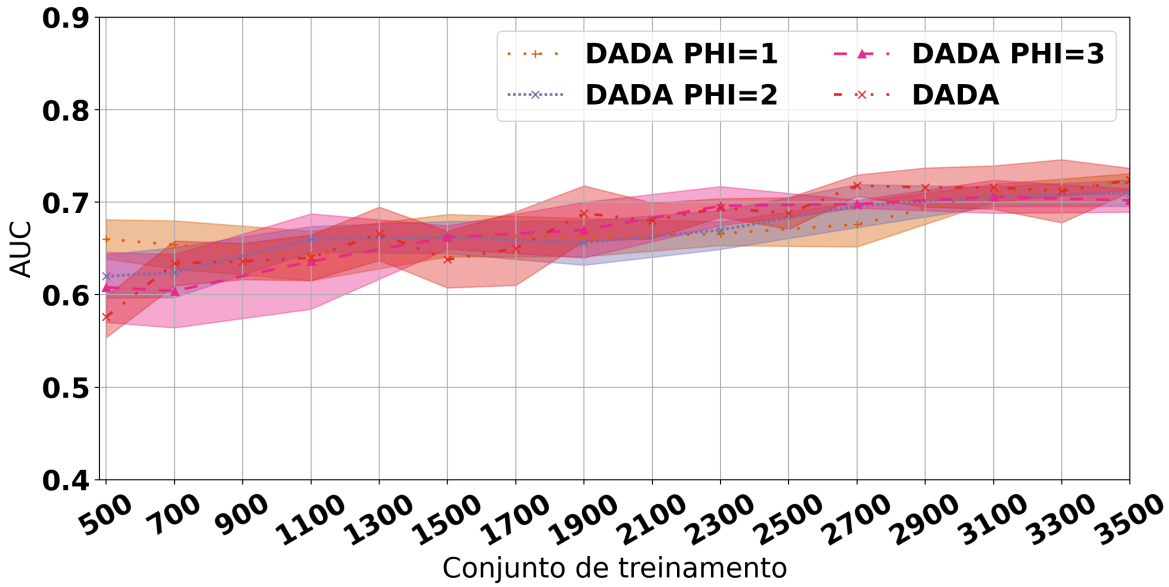


Figura 7.17: Níveis de AUC obtidos pela rede ResNet50V2, com seleção de *patches* realizados pelas redes reduzidas pelo NAR, na abordagem MC Dropout.

casos. Uma diferença em relação à Inception é que as curvas de todos os níveis de redução aplicáveis à ResNet ficaram próximas umas das outras, comportamento que só havia sido observado com o uso de DA.

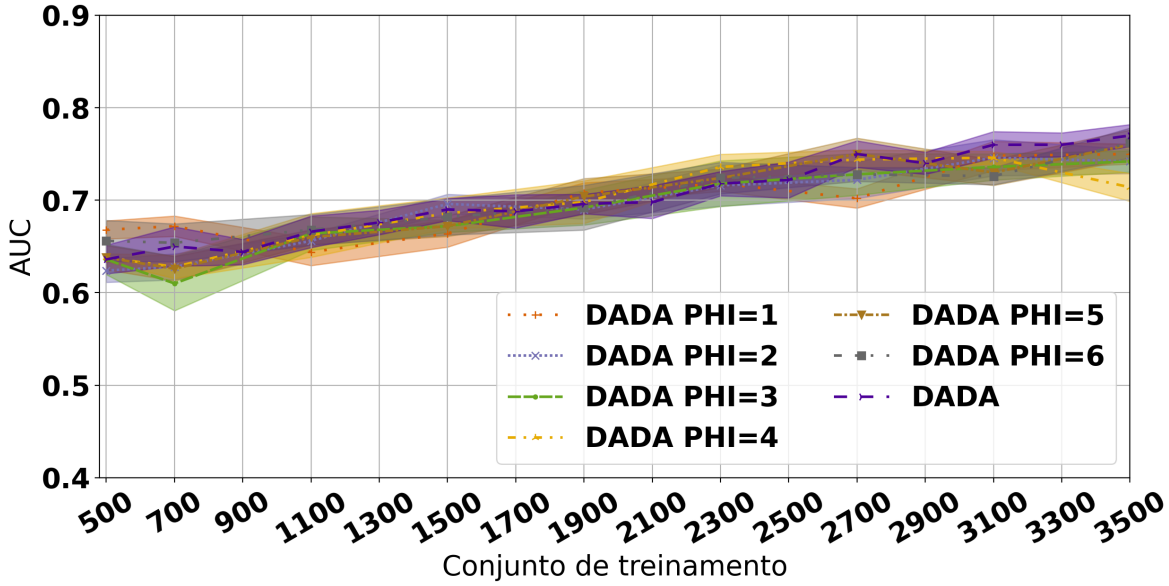


Figura 7.18: Níveis de AUC obtidos pela rede Xception, com seleção de *patches* realizados pelas redes reduzidas pelo NAR, na abordagem MC Dropout.

Assim como no caso da ResNet50 V2, a rede Xception apresentou comportamento

similar à Inception V4 com o uso de *data augmentation* no que diz respeito à proximidade das curvas de AUC utilizando-se o NAR (Figura 7.18). O DADA com Xception atingiu uma média de 0,70, que foi igualado pelo conjunto DADA/NAR com índices $\phi = 4, 5\ e\ 6$. Utilizando-se os fatores de redução $\phi = 2\ e\ 3$ a solução obteve 0,69 de média, demonstrando pequena variação frente aos demais valores. Esses resultados demonstram que para essa rede o impacto em AUC do uso do NAR é marginal.

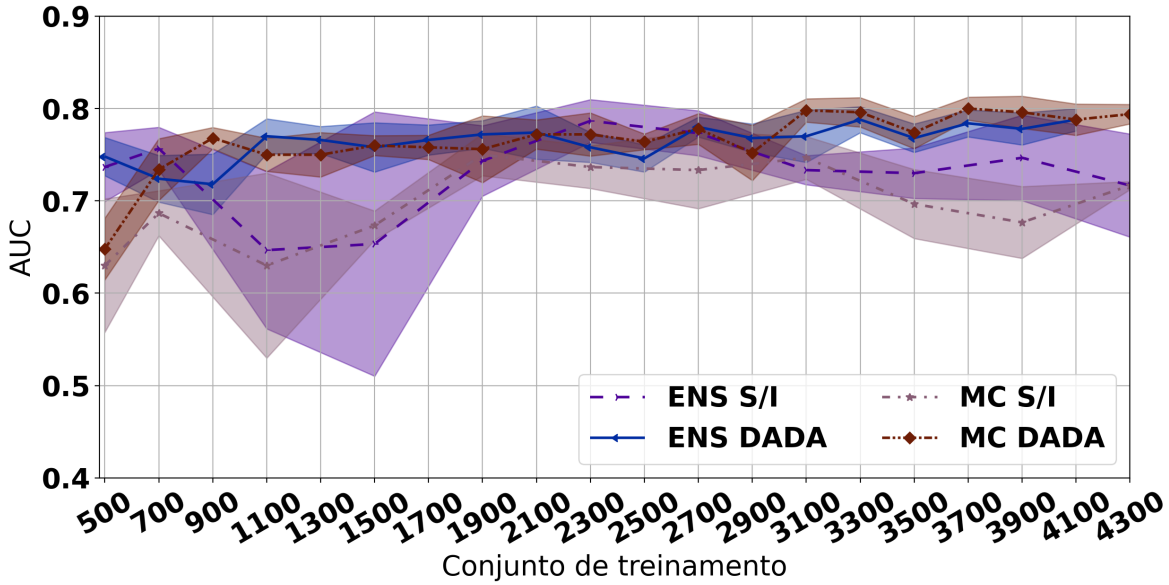


Figura 7.19: Níveis de AUC obtidos pela rede Inception V4, com seleção de *patches* realizados pela rede SmallNet, nas abordagens MC Dropout e por Ensembles.

A utilização de modelos simplificados pelo NAR junto ao DADA tem se mostrado viável, todavia, resta avaliar se essas simplificações são melhores do que o uso de modelos totalmente distintos da rede alvo, porém de muito baixo custo. Para esse fim, foi elaborada uma rede pequena, batizada de *SmallNet*, para avaliar a qualidade da seleção de *patches* produzida por essa rede, quando usados para treinar a rede alvo Inception V4. Essa rede é composta de 2 blocos contendo 2 camadas convolucionais e uma camada de *MaxPooling*, com um bloco final para extração de características com 1 camada convolucional. No total, 5 camadas convolucionais formam a rede, que tem 413.250 parâmetros. Os resultados do uso dessa rede junto ao DADA são mostrados na Figura 7.19.

A figura mostra o resultado do uso da rede SmallNet nas abordagens MC Dropout (MC S/I) e por Ensembles (ENS S/I). No primeiro caso, a média de AUC atingiu 0,70, enquanto com Ensembles foi de 0,73. Esses valores são compatíveis aos obtidos pelo DADA/NAR utilizando-se o nível de redução $\phi = 6$ (ver Tabela 7.10). Interessante notar também que, apesar das médias serem próximas das obtidas pela versão mais agressiva do NAR, os intervalos de confiança são consideravelmente mais amplos, mostrando uma

variabilidade expressiva dos resultados. Os experimentos corroboram a hipótese que redes muito distintas da rede alvo possuem tendência a produzir uma seleção de *patches* pior que aquela feita pela própria rede alvo.

Considerações Sobre o Uso do NAR

A análise conjunta dos resultados obtidos com o uso do NAR junto ao DADA permitem construir argumentos robustos acerca da viabilidade da utilização de redes mais simples para executar o processo de cálculo de incertezas e geração dos agrupamentos de *patches* demandados pelo DADA. Apesar de que redes muito simples poderiam ser usadas para esse fim, o desempenho tende a ser pior do que o obtido por redes próximas à original.

Os ganhos de tempo de processamento obtidos pelo NAR justificam seu uso e mostram que a escolha original de qual CNN usar junto ao DADA deve ser, primeiramente, com relação à sua capacidade de classificação dentro do contexto da aplicação a que se destina, sendo possível compensar eventuais custos computacionais com o uso do NAR. Dentro do escopo da aplicação motivadora aqui estudada, mostrou-se como a melhor solução na maioria dos experimentos o uso do MC Dropout DADA/NAR com *data augmentation* e a rede reduzida $\phi = 4$. Nessa configuração os ganhos de tempo em relação ao DADA foram de 3,06 $\times$.

7.3 Impacto dos Parâmetros do DADA em seu Desempenho

DADA introduz novos parâmetros em relação às demais abordagens, como por exemplo a dimensão das características produzidas pelo PCA, número de *clusters* e o tamanho e frequência de regeneração do *subpool*. Esses parâmetros são avaliados agora, considerando o ENS DADA como referência.

O impacto da dimensionalidade das características após redução pelo PCA, utilizadas na formação de clusters para o aprendizado ativo, é mostrado na Figura 7.20. Como esperado, o número de características impacta o desempenho do DADA e, apesar dos resultados com a dimensão de 50 serem equivalentes àqueles obtidos com 150 características, a partir de 450 e além, o problema da dimensionalidade [188] se mostra claramente, resultando em níveis de AUC mais baixos quando comparado aos obtidos com conjuntos de características menores. É importante destacar que a rede InceptionV4 [183], usada em nossa aplicação de classificação de LITs produz 1.536 características na saída da última camada convolucional. Isso mostra que a redução de dimensionalidade, aqui executada através do PCA, é necessária e gera melhorias de AUC importantes para o AL.

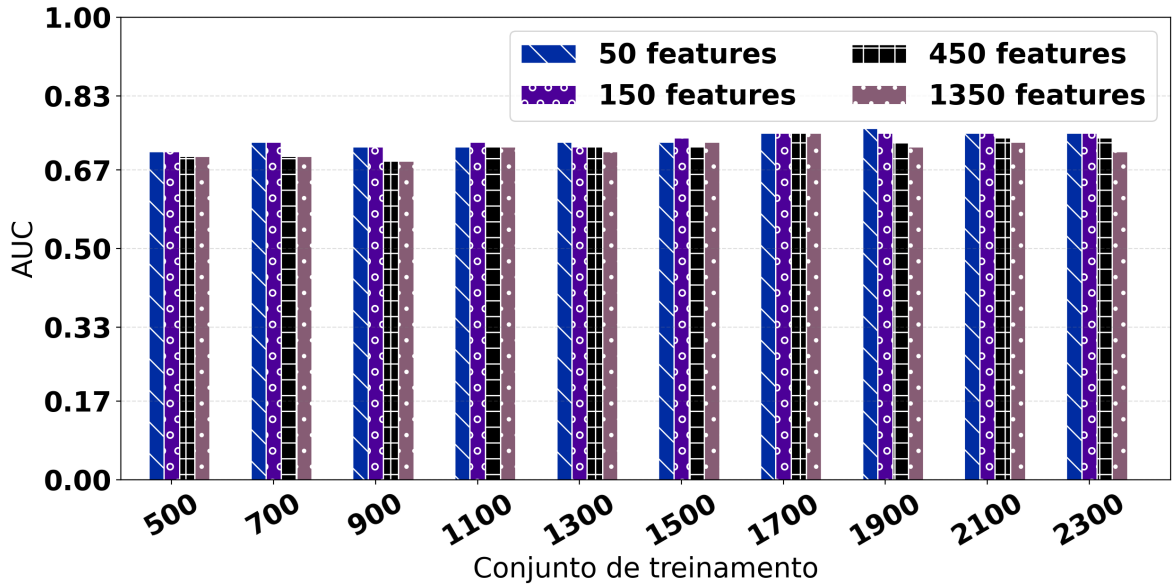


Figura 7.20: Análise de dimensionalidade das características gerados para ENS DADA.

O número de *clusters* formados a partir das características [189] também foi avaliado (Figura 7.21). Um número mais elevado de *clusters* pode ter um efeito positivo à medida que o número de imagens no conjunto de treinamento cresce, o que pode estar relacionado a um aumento nos grupos de características presentes. Todavia, as oportunidades de ganho são limitadas pelo número de tipos de tecidos de fato presentes nas amostras. Considerando o uso de um número de conjuntos igual ao tamanho do conjunto de aquisição (200 nos experimentos realizados), na verdade prejudica o desempenho. A partir da Figura 7.21 fica claro que utilizar o número de 20 *clusters* é benéfico nos primeiros estágios do processo de aprendizado ativo, ajudando a acelerar o desempenho do treinamento com pequena quantidade de *patches*. Na média, as configurações com 20, 40 e 80 *clusters* alcançaram AUC de 0,735, 0,739 e 0,732, nessa ordem, enquanto a versão com 200 *clusters* obteve 0,697. Dado o exposto, configuração utilizada nos resultados apresentados nas seções anteriores foi de 20 *clusters*.

A influência do tamanho do *subpool* para o desempenho do DADA é parcialmente demonstrada na Figura 7.3b e também nos gráficos da Figura 7.22. Como já citado, o tamanho de *subpool*, com 2.000 *patches* obteve os melhores resultados sobre as demais versões. A variante com *subpool* de 1.000 *patches* apresentou AUC médio de 0,71, sendo superada pela configuração do ENS DADA, que obteve 0,74. O segundo melhor resultado foi alcançado com o conjunto de 10.000 *patches*, com 0,73.

A frequência de regeneração do *subpool* foi avaliada e os resultados constam da Figura 7.23. Três diferentes configurações foram analisadas, com a regeneração do *subpool* a cada 2, 3 e 4 iterações do aprendizado ativo, resultando, respectivamente, em 5, 3 e 2

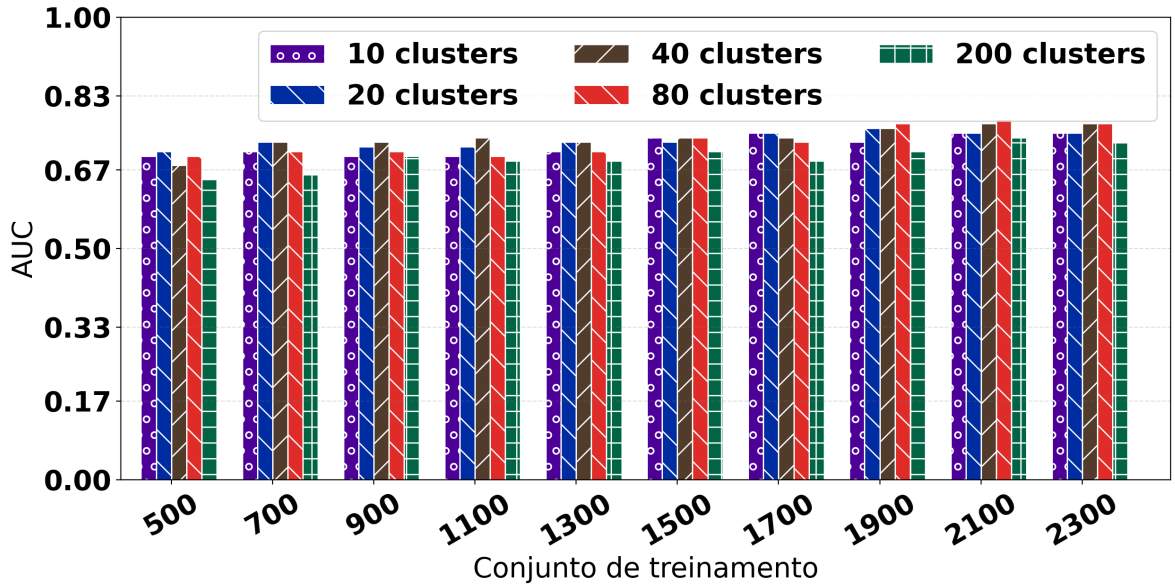


Figura 7.21: Análise do # de clusters para DDBALD (2k).

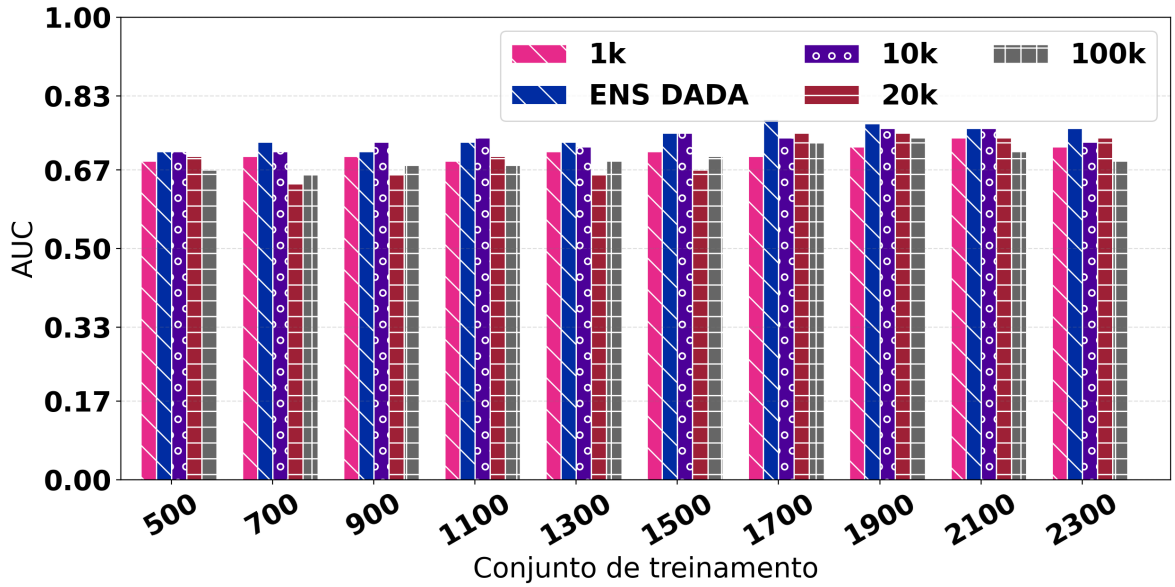


Figura 7.22: Análise do tamanho do *subpool* para ENS DADA.

subpools, utilizados para alimentar 10 iterações de aquisição. Os resultados demonstram que, para esse parâmetro, o desempenho foi afetado apenas marginalmente, com a maior diferença observada após a aquisição de 1.700 e 2.300 *patches*, em ambos os casos por uma margem de AUC de 0,003.

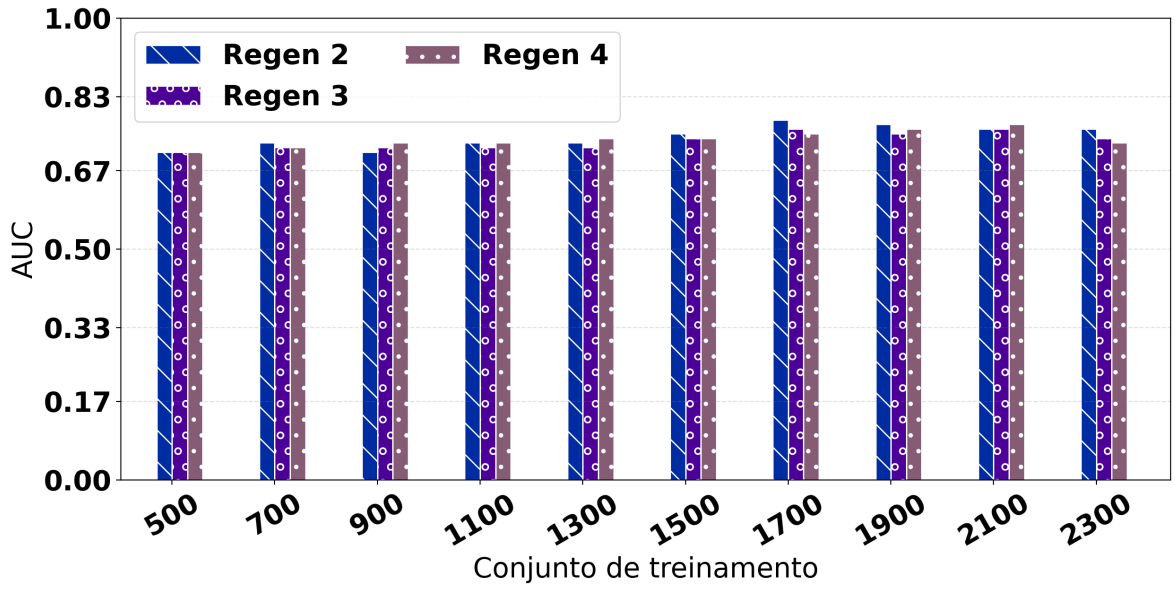


Figura 7.23: Frequência de regeneração do *subpool* para ENS DADA.

Capítulo 8

Conclusão

Essa tese apresentou o Diversity-Aware Data Aquisition (DADA), uma solução de aprendizado ativo eficaz para selecionar *patches* que devem receber anotações por um especialista e, assim, formar uma base de dados de treinamento para CNNs. A seleção realizada pelo DADA busca produzir um conjunto de treinamento de tamanho mínimo, de modo a reduzir os custos associados ao processo de anotação. Ao mesmo tempo em que deve ser minimizado o tamanho desse conjunto, as imagens nele contidas devem ser tais que uma CNN alvo possa ser treinada de forma a produzir um desempenho de classificação esperado ou aceitável dentro do escopo da aplicação.

O foco da solução desenvolvida é a identificação de áreas de tecido com a presença de LITs, visto que esse achado médico já foi correlacionado a maior sobrevida em pacientes com diversos tipos de tumor (ver Seção 5.1). Tendo esse foco, o DADA foi elaborado de maneira a procurar explorar as características desse tipo de imagem e como conjuntos delas podem ser melhores ou piores para se treinar uma CNN. A inspiração para o método de seleção usado no DADA advém da análise das soluções já existentes e também do estudo das imagens que são foco desse trabalho.

Essa estratégia passou por um estágio de validação inicial, sendo aplicada a uma base de dados simples mas muito utilizada na literatura, a base MNIST. Os resultados demonstraram que o uso do DADA pode contribuir para uma seleção de pontos de treinamento de maneira mais eficiente que as demais técnicas avaliadas (Ensemble e Monte Carlo), especialmente nas iterações iniciais, atingindo níveis de acurácia mais elevados com menor quantidade de imagens. Esses resultados incentivaram a continuidade dos testes do DADA, demonstrando estar pronto para avaliações mais profundas sobre a base de dados de LITs.

Ao contrário dos resultados positivos vistos com os testes sobre a base MNIST, observou-se que as técnicas clássicas disponíveis na literatura, com o uso apenas de métricas de incertezas, não seriam suficientes para produzir um bom conjunto de treinamento sobre

a aplicação motivadora. Assim, soluções como o Monte Carlo Dropout [118] ou Ensembles para aprendizado ativo [117] não produziam conjuntos de treinamento pequenos e representativos o suficiente para treinar uma CNN de forma adequada sobre a base de LITs. Soluções como o CoreSet [92], que busca as imagens mais representativas dentre um conjunto de exemplos não rotulados, também não se mostrou suficiente para trazer o desempenho de classificação do modelo para níveis aceitáveis dentro do contexto da medicina.

A composição entre representatividade e incertezas foi a solução encontrada e usada pelo DADA. As métricas de incerteza buscam identificar as imagens próximas ao limite de decisão dos classificadores, de modo que, ao introduzí-las no conjunto de treinamento, esses modelos tendem a expandir sua capacidade de reconhecimento e classificação de pontos "difíceis". Ao mesmo tempo, inserir imagens representativas frente ao conjunto não rotulado adiciona capacidade de generalização ao classificador.

O DADA então utiliza o agrupamento de *patches* em conjuntos semelhantes e executa uma seleção de pontos desses conjuntos, com um algoritmo próprio que se utiliza dos níveis de incerteza para identificar a quantidade de pontos que serão extraídos de cada conjunto. Ao escolher obrigatoriamente *patches* de diferentes conjuntos, o DADA procura adicionar representatividade às imagens escolhidas e, ao mesmo tempo, seleciona as imagens com os maiores níveis de incerteza, com o objetivo de acelerar o aprendizado do modelo.

Os resultados obtidos com essa estratégia foram melhores que os possíveis com as soluções já listadas. Todavia, dado o duplo objetivo da solução, minimização do conjunto de treinamento e redução dos custos de anotação, observou-se que o tempo de processamento demandado ainda era grande para o uso do DADA na prática. A aplicação de *subpools* dinâmicos, com regeneração periódica reduziu esse tempo à metade ao se utilizar a abordagem por Ensembles e, no caso da abordagem do MC Dropout, fez com que ela se tornasse não só viável como se tornou a opção mais rápida, sendo 4,52× mais veloz que o ENS DADA com *pool* fixo de 100.000 *patches*.

Mesmo com as modificações e introdução do *subpool*, o tempo de execução para cada iteração pode ainda ser alto para algumas aplicações. Mesmo dispondo de *hardware* de alta capacidade de processamento, com um conjunto inicial de treinamento de 500 *patches* o ENS DADA ainda demanda cerca de 36 minutos para selecionar um novo conjunto para anotações e o MC DADA leva cerca de 17 minutos. Esses tempos vão aumentando à medida que cresce também o conjunto de treinamento. A fim de reduzir esse tempo ainda mais, foi desenvolvido o NAR, que tem aplicação direta sobre redes estruturadas em blocos, podendo também ser facilmente adaptado a redes com estrutura sequencial, conforme descrito na Seção 6.3.2.

Com o NAR, os tempos médios de cada iteração do DADA tiveram novas reduções

significativas. Os resultados obtidos mostram que a configuração mais eficaz para o DADA/NAR, no contexto da aplicação motivadora foco deste trabalho, é a combinação da abordagem MC Dropout e o uso de fator de redução $\phi = 4$, valor que traria ganhos de velocidade médios da ordem de 3,06×, com redução mínima ou nenhuma no desempenho de classificação. Com essa configuração as iterações do DADA possuem tempos menores que 10 minutos até que se atinja um conjunto de treinamento com 1.100 *patches*.

A aplicação do NAR permitiu ao DADA se aproximar de tempos de execução palpáveis para que seja utilizado na prática. Assim, foi desenvolvida uma interface gráfica a partir da adaptação do QuickAnnotator [165] para uso conforme o fluxo de trabalho definido pelo DADA. Essa interface permite a um *expert* visualizar o *patch* para o qual se necessita fornecer uma anotação, bem como as regiões de tecido ao seu redor, facilitando uma correta classificação pelo anotador. A interface possui navegação amigável, sendo de fácil uso através de qualquer navegador Web. Com ela será possível efetivamente comparar a qualidade das anotações e o custo de tempo em medições reais que vislumbra-se serem executadas em breve.

Considerando o estágio de desenvolvimento atingido, estão previstos entre os trabalhos futuros um aprofundamento dos estudos sobre a capacidade de generalização do DADA para outras aplicações e bases de dados. As características das imagens advindas de WSIs, juntamente com estudos experimentais, levaram ao desenvolvimento do DADA, com a definição de parâmetros como o número de *clusters* usado, o algoritmo de agrupamento, o número de características extraídas das imagens, entre outros. Assim, é importante avaliar o quanto essa configuração é diretamente aplicável a outros cenários ou o nível de adaptações que seriam necessárias para que o DADA apresente seu melhor resultado.

Além da classificação, o problema de segmentação demanda esforços de anotação ainda maiores. Dessa forma, outro trabalho futuro importante seria a adaptação do DADA para uso nesse tipo de problema e a verificação de sua capacidade de produzir conjuntos de treinamento otimizados. O sucesso da aplicação do DADA nesses cenários possibilitaria a criação de bases de dados que seriam fundamentais para pesquisas em diversos contextos, na área médica e também em campos diferentes, como sensoriamento remoto [190].

O NAR também é uma contribuição que merece maiores estudos em trabalhos futuros, sendo importante melhor avaliar o quanto essa técnica pode ser utilizada de maneira mais abrangente como uma solução de simplificação de CNNs, com o uso das versões reduzidas para efetivamente realizar tarefas de classificação de imagens, tanto no contexto médico como em outras áreas de aplicação. Vale ressaltar mais uma vez que, em seu desenvolvimento, não objetivou-se utilizar as redes reduzidas como classificadores, apenas que fossem ferramentas para os cálculos de incertezas e extração de características das

imagens para posteriormente agrupá-las em *clusters*.

Nessa linha, os estudos sobre o NAR pendem uma investigação mais profunda acerca da caracterização dos modelos reduzidos e o quanto eles seriam "próximos" dos originais com relação às incertezas produzidas, ou mesmo se seria possível desenvolver modelos de baixo custo, projetados especificamente para gerarem incertezas e agrupamentos aplicáveis ao DADA, tendo como alvo uma CNN mais robusta. Esses estudos poderiam implicar em reduções de tempo de iteração ainda maiores, tornando o DADA aplicável mesmo com o uso de *hardware* mais simples.

Essas evoluções seriam importantes para abrir caminhos para o uso de soluções de *deep learning* tanto para aplicações que sofrem com a falta de bases de dados de treinamento suficientes quanto para contextos de restrições de orçamento e de poder computacional, que são a regra na prática.

Referências

[1] Acemoglu, Daron e Pascual Restrepo: *Artificial intelligence, automation and work*. Relatório Técnico, National Bureau of Economic Research, 2018. 1

[2] Heer, Jeffrey: *Agency plus automation: Designing artificial intelligence into interactive systems*. Proceedings of the National Academy of Sciences, 116(6):1844–1850, 2019. 1

[3] *IBM Watson*, agosto 2022. `https://www.ibm.com/br-pt/watson-health`. 1

[4] Kong, Jun, Lee A.D. Cooper, Fusheng Wang, Jingjing Gao, George Teodoro, Lisa Scarpace, Tom Mikkelsen, Matthew J. Schniederjan, Carlos S. Moreno, Joel H. Saltz e Daniel J. Brat: *Machine-Based Morphologic Analysis of Glioblastoma Using Whole-Slide Pathology Images Uncovers Clinically Relevant Molecular Correlates*. PLOS One, November 2013. 1

[5] Hou, Le, Dimitris Samaras, Thasin Kurc, Yi Gao, James E. Davis e Joel H. Saltz: *Patch-based Convolutional Neural Network for Whole Slide Tissue Image Classification*. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, páginas 2424 – 2433, October 2016. 1

[6] LeCun, Yann, Yoshua Bengio e Geoffrey E. Hinton: *Deep Learning*, 2015. 2, 7

[7] Litjens, Geert, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen AWM van der Laak, Bram van Ginneken e Clara I Sánchez: *A survey on deep learning in medical image analysis*. Medical Image Analysis, 42:60–88, 2017. 2

[8] Wang, Sheng, Jiawen Yao, Zheng Xu e Junzhou Huang: *Subtype cell detection with an accelerated deep convolution neural network*. Em *International Conference on Medical Image Computing and Computer-Assisted Intervention*, páginas 640–648. Springer, 2016. 2

[9] Shen, Dinggang, Guorong Wu e Heung Il Suk: *Deep learning in medical image analysis*. Annual Review of Biomedical Engineering, 19:221–248, junho 2017. 2

[10] Sirinukunwattana, Korsuk, Shan E. Ahmed Raza, Yee Wah Tsang, David R. J. Snead, Ian A. Cree e Nasir M. Rajpoot: *Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images*. IEEE Transactions on Medical Imaging, 35(5):1196–1206, 2016. 2

[11] Tokunaga, Hiroki, Yuki Teramoto, Akihiko Yoshizawa e Ryoma Bise: *Adaptive weighting multi-field-of-view cnn for semantic segmentation in pathology.* Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 12597–12606, 2019. 2

[12] Gertych, Arkadiusz, Zaneta Swiderska-Chadaj, Zhaoxuan Ma, Nathan Ing, Tomasz Markiewicz, Szczepan Cierniak, Hootan Salemi, Samuel Guzman, Ann E. Walts e Beatrice S. Knudsen: *Convolutional neural networks can accurately distinguish four histologic growth patterns of lung adenocarcinoma in digital slides.* Scientific Reports, 9(1):1483, 2019. 2

[13] Wang, Shidan, Ruichen Rong, Donghan M Yang, Junya Fujimoto, Shirley Yan, Ling Cai, Lin Yang, Danni Luo, Carmen Behrens, Edwin R. Parra, Bo Yao, Lin Xu, Tao Wang, Xiaowei Zhan, Ignacio I. Wistuba, John Minna, Yang Xie e Guanghua Xiao: *Computational staining of pathology images to study the tumor microenvironment in lung cancer.* Cancer Research, 2020, ISSN 0008-5472. `https://cancerres.aacrjournals.org/content/early/2020/01/08/0008-5472.CAN-19-1629`. 2

[14] Roh, Yuji, Geon Heo e Steven Euijong Whang: *A survey on data collection for machine learning: a big data-ai integration perspective.* IEEE Transactions on Knowledge and Data Engineering, 2019. 2, 18

[15] Amgad, Mohamed, Habiba Elfandy, Hagar Hussein, Lamees A Atteya, Mai A T Elsebaie, Lamia S Abo Elnasr, Rokia A Sakr, Hazem S E Salem, Ahmed F Ismail, Anas M Saad, Joumana Ahmed, Maha A T Elsebaie, Mustafijur Rahman, Inas A Ruhban, Nada M Elgazar, Yahya Alagha, Mohamed H Osman, Ahmed M Alhusseiny, Mariam M Khalaf, Abo Alela F Younes, Ali Abdulkarim, Duaa M Younes, Ahmed M Gadallah, Ahmad M Elkashash, Salma Y Fala, Basma M Zaki, Jonathan Beezley, Deepak R Chittajallu, David Manthey, David A Gutman e Lee A D Cooper: *Structured crowdsourcing enables convolutional segmentation of histology images.* Bioinformatics, 35(18):3461–3467, fevereiro 2019, ISSN 1367-4803. `https://doi.org/10.1093/bioinformatics/btz083`. 2, 18

[16] Deng, Jia, Wei Dong, Richard Socher, Li Jia Li, Kai Li e Fei Fei Li: *Imagenet: A large-scale hierarchical image database.* Em *2009 IEEE Conference on Computer Vision and Pattern Recognition*, páginas 248–255. Ieee, 2009. 2, 16

[17] Ørting, Silas, Andrew Doyle, Matthias Hirth Arno van Hilten, Oana Inel, Christopher R Madan, Panagiotis Mavridis, Helen Spiers e Veronika Cheplygina: *A survey of crowdsourcing in medical image analysis.* arXiv preprint arXiv:1902.09159, 2019. 2, 18

[18] Kim, Edward, Sai Lakshmi Deepika Mente, Andrew Keenan e Vijay Gehlot: *Digital pathology annotation data for improved deep neural network classification.* Em *Medical Imaging 2017: Imaging Informatics for Healthcare, Research, and Applications*, volume 10138, página 101380D, 2017. 2, 18

[19] Hou, Le, Ayush Agarwal, Dimitris Samaras, Tahsin M Kurc, Rajarsi R Gupta e Joel H Saltz: *Robust histopathology image analysis: to label or to synthesize?* Em

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, páginas 8533–8542, 2019. 2, 18

[20] Hossain, Md Shamim e Nazmus Sakib: *Renal cell cancer nuclei segmentation from histopathology image using synthetic data.* Em *2020 16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA)*, páginas 236–241, 2020. 2, 18

[21] Sankaranarayanan, Swami, Yogesh Balaji, Arpit Jain, Ser Nam Lim e Rama Chellappa: *Learning from synthetic data: Addressing domain shift for semantic segmentation.* Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 3752–3761, 2018. 2

[22] Cohn, David A, Zoubin Ghahramani e Michael I Jordan: *Active learning with statistical models.* Journal of Artificial Intelligence Research, 4:129–145, 1996. 3, 19

[23] Carse, Jacob e Stephen McKenna: *Active learning for patch-based digital pathology using convolutional neural networks to reduce annotation costs.* Em *European Congress on Digital Pathology*, páginas 20–27. Springer, 2019. 3, 53

[24] Rączkowski, Łukasz, Marcin Możejko, Joanna Zambonelli e Ewa Szczurek: *Ara: accurate, reliable and active histopathological image classification framework with bayesian deep learning.* Scientific Reports, 9(1):1–12, 2019. 3, 53

[25] Meirelles, André L.S., Tahsin Kurc, Joel Saltz e George Teodoro: *Effective active learning in digital pathology: A case study in tumor infiltrating lymphocytes.* Computer Methods and Programs in Biomedicine, 220:106828, 2022, ISSN 0169-2607. `https://www.sciencedirect.com/science/article/pii/S0169260722002103`. 5

[26] Meirelles, André L. S., Tahsin Kurc, Jun Kong, Renato Ferreira, Joel Saltz e George Teodoro: *Building efficient cnn architectures for histopathology images analysis: A case-study in tumor-infiltrating lymphocytes.* Frontiers in Medicine, 9:1604, June 2022. 5

[27] Michie, Donald, David J Spiegelhalter, CC Taylor *et al.*: *Machine learning.* Neural and Statistical Classification, 13(1994):1–298, 1994. 6

[28] Hastie, Trevor, Robert Tibshirani e Jerome Friedman: *The elements of statistical learning: data mining, inference, and prediction.* Springer Science & Business Media, 2009. 6, 7, 9, 11

[29] Quinlan, J. Ross: *Induction of decision trees.* Machine Learning, 1(1):81–106, 1986. 6

[30] Rokach, Lior: *Decision Forest: Twenty Years of Research.* Information Fusion, 27:111–125, 2016. 6, 10

[31] Kleinbaum, David G, K Dietz, M Gail, Mitchel Klein e Mitchell Klein: *Logistic Regression.* New York: Springer-Verlag, 2002. 7

[32] Peng, Chao Ying Joanne, Kuk Lida Lee e Gary M Ingersoll: *An Introduction to Logistic Regression Analysis and Reporting.* The Journal of Educational Research, 96(1):3–14, 2002. 7

[33] Noble, William S: *What is a support vector machine?* Nature Biotechnology, 24(12):1565–1567, 2006. 7, 11

[34] McCulloch, Warren S e Walter Pitts: *A logical calculus of the ideas immanent in nervous activity.* The Bulletin of Mathematical Biophysics, 5(4):115–133, 1943. 7, 11

[35] Rumelhart, David E, Geoffrey E Hinton, James L McClelland *et al.*: *A General Framework for Parallel Distributed Processing.* Parallel Distributed Processing: Explorations in the Microstructure of Cognition, 1(45-76):26, 1986. 7, 11

[36] Domingos, Pedro e Michael Pazzani: *On the optimality of the simple bayesian classifier under zero-one loss.* Machine Learning, 29(2-3):103–130, 1997. 7

[37] Jordan, Michael I e Tom M Mitchell: *Machine learning: Trends, perspectives, and prospects.* Science, 349(6245):255–260, 2015. 7

[38] Soffer, Shelly, Avi Ben-Cohen, Orit Shimon, Michal Marianne Amitai, Hayit Greenspan e Eyal Klang: *Convolutional neural networks for radiologic images: a radiologist's guide.* Radiology, 290(3):590–606, 2019. x, 7

[39] Kolles, Harry, Aldo v Wangenheim, Giles H. Vince, Isolde Niedermayer e Wolfgang Feiden: *Automated grading of astrocytomas based on histomorphometric analysis of Ki-67 and Feulgen stained paraffin sections. Classification results of neuronal networks and discriminant analysis.* Analytical Cellular Pathology, 8:101–116, 1994. 8

[40] Baeg, S e N Kehtarnavaz: *Texture based classification of mass abnormalities in mammograms.* Computer-Based Medical Systems, 2000. CBMS 2000. Proceedings. 13th IEEE Symposium on, June 2000. 8

[41] Barker, Jocelyn, Assaf Hoogi, Adrien Depeursinge e Daniel L. Rubin: *Automated classification of brain tumor type in whole-slide digital pathology images using local representative tiles.* Medica Image Analysis, 30:60–71, 2016. 8

[42] Boureau, Y Lan, Jean Ponce e Yann LeCun: *A theoretical analysis of feature pooling in visual recognition.* Em *International Conferece on Machine Learning, In Proc. of*, Haifa, Israel, 2010. 8

[43] Hafner, M., A. Gangl, M. Liedlgruber, A. Uhl e A. Vecsei and. F Wrba: *Combining Gaussian Markov Random Fields with the Discrete-Wavelet Transform for Endoscopic Image Classification.* Digital Signal Processing, 2009 16th International Conference on, July 2009. 8

[44] Netzer, Yuval, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu e Andrew Y Ng: *Reading digits in natural images with unsupervised feature learning.* Relatório Técnico, Google Inc., 2011. 8

[45] Kong, Jun, Lee A.D. Cooper, Fusheng Wang, Candance Chisolm, Carlos S. Moreno, Thasin Kurc, Patrick Widener, Daniel J. Brat e Joel H. Saltz: *A Comprehensive Framework For Classification Of Nuclei In Digital Microscopy Imaging: An Application To Diffuse Gliomas*. Em *Proc IEEE Int Symp Biomed Imaging*, páginas 2128–2131, March 2011. 8

[46] Saeys, Yvan, Iñaki Inza e Pedro Larrañaga: *A review of feature selection techniques in bioinformatics*. BMC Bioinformatics, 23(19):2507–2517, August 2007. 8

[47] Borque, A., G. Sanz, C. Allepuz, L. Plaza, P. Gil e L.A. Rioja: *The use of neural networks and logistic regression analysis for predicting pathological stage in men undergoing radical prostatectomy: a population based study*. The Journal of Urology, 166(5):1672–1678, 2001. 9

[48] Han, Misop, Peter B Snow, Jeffrey M Brandt e Alan W Partin: *Evaluation of artificial neural networks for the prediction of pathologic stage in prostate carcinoma*. Cancer: Interdisciplinary International Journal of the American Cancer Society, 91(S8):1661–1666, 2001. 9

[49] Sharma, Himani e Sunil Kumar: *A survey on decision tree algorithms of classification in data mining*. International Journal of Science and Research (IJSR), 5(4):2094–2097, 2016. 9

[50] Decaestecker, Christine, Isabelle Camby, Myriam Remmelink, Nathalie Nagy, Michel Petein, Jean Lambert Pasteels, Philippe Van Ham, Isabelle Salmon e Robert Kiss: *Decision tree induction: a useful tool for assisted diagnosis and prognosis in tumor pathology*. Laboratory Investigation: A Journal of Technical Methods and Pathology, 76(6):799, 1997. 9

[51] Kuo, Wen Jia, Ruey Feng Chang, Dar Ren Chen e Cheng Chun Lee: *Data mining with decision trees for diagnosis of breast tumor in medical ultrasonic images*. Breast Cancer Research and Treatment, 66(1):51–57, 2001. 9, 10

[52] Naik, Janki e Sagar Patel: *Tumor detection and classification using decision tree in brain mri*. International Journal of Computer Science and Network Security, 14(6):87, 2014. 9, 10

[53] Breiman, Leo, Jerome H. Friedman, Richard A. Olshen e Charles J. Stone Stone: *Classification and Regression Trees*. Routledge, 1st edição, October 1984. 10

[54] Quinlan, J Ross: *C4. 5: programs for machine learning*. Elsevier, 2014. 10

[55] Quinlan, J Ross: *Data mining tools see5 and c5. 0*, 2004. 10

[56] Freund, Yoav e Robert E Schapire: *Experiments with a new boosting algorithm*. Em *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, volume 96, páginas 148–156. Citeseer, 1996. 10

[57] Breiman, Leo: *Random forests*. Machine Learning, 45(1):5–32, 2001. 10

[58] Bhatia, Sumit, Praveen Prakash e G.N. Pillai: *Svm based decision support system for heart disease classification with integer-coded genetic algorithm to select critical features.* Em *Proceedings of the World Congress on Engineering and Computer Science*, páginas 34–38, 2008. 11

[59] Corredor, Germán, Xiangxue Wang, Cheng Lu, Vamsidhar Velcheti, Eduardo Romero e Anant Madabhushi: *A watershed and feature-based approach for automated detection of lymphocytes on lung cancer images.* Em *Medical Imaging 2018: Digital Pathology*, volume 10581, página 105810R. International Society for Optics and Photonics, 2018. 11

[60] Widrow, Bernard e Marcian E Hoff: *Adaptive switching circuits.* Relatório Técnico, Stanford University (CA) Stanford Electronics Labs, 1960. 11

[61] Arbach, Lina, Alan Stolpen e Joseph M. Reinhardt: *Classification of breast MRI lesions using a backpropagation neural network (BNN).* Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on, April 2004. 12

[62] Das, Resul, Ibrahim Turkoglu e Abdulkadir Sengur: *Effective diagnosis of heart disease through neural networks ensembles.* Expert Systems with Applications, 36(4):7675–7680, 2009. 12, 13

[63] Sanger, Terence D: *Optimal unsupervised learning in a single-layer linear feedforward neural network.* Neural Networks, 2(6):459–473, 1989. 13

[64] Bebis, George e Michael Georgiopoulos: *Feed-forward neural networks.* IEEE Potentials, 13(4):27–31, 1994. 13

[65] Tamura, Shin'ichi e Masahiko Tateishi: *Capabilities of a four-layered feedforward neural network: four layers versus three.* IEEE Transactions on Neural Networks, 8(2):251–255, 1997. 13

[66] Hecht-Nielsen, Robert: *Theory of the backpropagation neural network.* Em *Neural Networks for Perception*, páginas 65–93. Elsevier, 1992. 13, 64

[67] Cybenko, George: *Approximation by superpositions of a sigmoidal function.* Mathematics of Control, Signals and Systems, 2(4):303–314, 1989. 13

[68] Simonyan, Karen e Andrew Zisserman: *Very Deep Convolutional Networks for Large-Scale Image Recognition.* Em *ICLR 2015*. arXiv:1409.1556, 2014. 13, 70

[69] Rumelhart, David E, Geoffrey E Hinton e Ronald J Williams: *Learning internal representations by error propagation.* Relatório Técnico, California Univ San Diego La Jolla Inst for Cognitive Science, 1985. 13

[70] Rumelhart, David E, Geoffrey E Hinton e Ronald J Williams: *Learning representations by back-propagating errors.* Nature, 323(6088):533–536, 1986. 13

[71] LeCun, Yann, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard e Lawrence Jackel: *Handwritten Digit Recognition with a Back-propagation Network*. Em Touretzky, David S. (editor): *Advances in Neural Information Processing Systems 2*, páginas 396–404. Morgan Kaufmann Publishers Inc., 1990. 14

[72] Raina, Rajat, Anand Madhavan e Andrew Y Ng: *Large-scale deep unsupervised learning using graphics processors*. Em *Proceedings of the 26th Annual International Conference on Machine Learning*, páginas 873–880, 2009. 14

[73] LeCun, Yann, Léon Bottou, Yoshua Bengio e Patrick Haffner: *Gradient-based learning applied to document recognition*. Em *Proceedings of the IEEE*, volume 86, páginas 2278–2324. IEEE, 1998. 14, 34, 35, 72

[74] Fukushima, Kunihiko e Sei Miyake: *Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition*. Em *Competition and Cooperation in Neural Nets*, páginas 267–285. Springer, 1982. 14

[75] Wan, Tao, Jiajia Cao, Jianhui Chen e Zengchang Qin: *Automated grading of breast cancer histopathology using cascaded ensemble with combination of multi-level image features*. Neurocomputing, 229:34–44, March 2017. 16

[76] Xu, Yan, Zhipeng Xia, Yuqing Ai, Fang Zhang, Maode Lai e Eric I Chao Chang: *Deep Convolutional Activation Features for Large Scale Brain Tumor Histopathology Image Classification and Segmentation*. Em *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, Australia, April 2015. IEEE. 16

[77] Xu, Yan, Zhipeng Xia, Liang Bo Wang, Yuqing Ai, Fang Zhang, Maode Lai e Eric I Chao Chang: *Large scale tissue histopathology image classification, segmentation, and visualization via deep convolutional activation features*. BMC Bioinformatics, 18(281), May 2017. 16

[78] Mikołajczyk, Agnieszka e Michał Grochowski: *Data augmentation for improving deep learning in image classification problem*. Em *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, páginas 117–122. IEEE, IEEE, 2018. 17

[79] Inel, Oana, Khalid Khamkham, Tatiana Cristea, Anca Dumitrache, Arne Rutjes, Jelle van der Ploeg, Lukasz Romaszko, Lora Aroyo e Robert Jan Sips: *Crowdtruth: Machine-human computation framework for harnessing disagreement in gathering annotated data*. Em *International Semantic Web Conference*, páginas 486–504. Springer, 2014. 18

[80] Arganda-Carreras, Ignacio, Srinivas C Turaga, Daniel R Berger, Dan Cireşan, Alessandro Giusti, Luca M Gambardella, Jürgen Schmidhuber, Dmitry Laptev, Sarvesh Dwivedi, Joachim M Buhmann *et al.*: *Crowdsourcing the creation of image segmentation algorithms for connectomics*. Frontiers in Neuroanatomy, 9:142, 2015. 18

[81] Li, Shan e Weihong Deng: *Reliable crowdsourcing and deep locality-preserving learning for unconstrained facial expression recognition.* IEEE Transactions on Image Processing, 28(1):356–370, 2018. 18

[82] Bug, Daniel, Gregor Nickel, Anne Grote, Friedrich Feuerhake, Eva Oswald, Julia Schüler e Dorit Merhof: *Image quilting for histological image synthesis.* Em *Bildverarbeitung für die Medizin 2020*, páginas 322–327. Springer, 2020. 18

[83] Gupta, Ankush, Andrea Vedaldi e Andrew Zisserman: *Synthetic data for text localisation in natural images.* Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 2315–2324. IEEE, 2016. 18

[84] Efros, Alexei A e William T Freeman: *Image quilting for texture synthesis and transfer.* Em *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, páginas 341–346, 2001. 18

[85] MacKay, David JC: *Information-based objective functions for active data selection.* Neural Computation, 4(4):590–604, 1992. 19

[86] Cohn, David A: *Neural network exploration using optimal experiment design.* Em *Advances in Neural Information Processing Systems*, páginas 679–686, 1994. 19

[87] Dasgupta, Sanjoy: *Analysis of a greedy active learning strategy.* Advances in Neural Information Processing Systems, 17:337–344, 2005. 19

[88] Settles, Burr: *Active learning literature survey.* Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009. 20, 36

[89] Baum, Eric B e Kenneth Lang: *Query learning can work poorly when a human oracle is used.* Em *International Joint Conference on Neural Networks*, volume 8, página 8, 1992. 20

[90] Li, Mingkun e Ishwar K Sethi: *Confidence-based active learning.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(8):1251–1261, 2006. 21

[91] Kendall, Alex e Yarin Gal: *What uncertainties do we need in bayesian deep learning for computer vision?* Em *Advances in Neural Information Processing Systems*, páginas 5574–5584, 2017. 21

[92] Sener, Ozan e Silvio Savarese: *Active learning for convolutional neural networks: A core-set approach.* International Conference on Learning Representations (ICLR), 2018. 21, 33, 35, 36, 37, 38, 104

[93] Yang, Yi, Zhigang Ma, Feiping Nie, Xiaojun Chang e Alexander G Hauptmann: *Multi-class active learning by uncertainty sampling with diversity maximization.* International Journal of Computer Vision, 113(2):113–127, 2015. 21

[94] Li, Xin e Yuhong Guo: *Adaptive active learning for image classification.* Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 859–866, 2013. 21, 25, 26, 27, 28

[95] Zhu, Jingbo, Huizhen Wang, Benjamin K Tsou e Matthew Ma: *Active learning with sampling by uncertainty and density for data annotations*. IEEE Transactions on Audio, Speech, and Language Processing, 18(6):1323–1331, 2009. 21

[96] Yuan, Jin, Xingxing Hou, Yaoqiang Xiao, Da Cao, Weili Guan e Liqiang Nie: *Multi-criteria active deep learning for image classification*. Knowledge-Based Systems, 172:86–94, 2019. 21, 39

[97] Fu, Yifan, Xingquan Zhu e Bin Li: *A survey on instance selection for active learning*. Knowledge and Information Systems, 35(2):249–283, 2013. 21

[98] Shannon, Claude E: *A mathematical theory of communication*. The Bell System Technical Journal, 27(3):379–423, 1948. 22

[99] Gal, Yarin e Zoubin Ghahramani: *Dropout as a bayesian approximation: Representing model uncertainty in deep learning*. Em *International Conference on Machine Learning*, páginas 1050–1059, 2016. 22, 30

[100] Gal, Yarin: *Uncertainty in deep learning*. Tese de Doutoramento, PhD thesis, University of Cambridge, 2016. 22, 28, 29, 30, 34

[101] Freeman, Linton C: *Elementary applied statistics: for students in behavioral science*. John Wiley & Sons, December 1965. 23

[102] Houlsby, Neil, Ferenc Huszár, Zoubin Ghahramani e Máté Lengyel: *Bayesian active learning for classification and preference learning*. arXiv preprint arXiv:1112.5745, 2011. 23

[103] Tong, Simon e Daphne Koller: *Support vector machine active learning with applications to text classification*. Journal of Machine Learning Research, 2(Nov):45–66, 2001. 25

[104] Nguyen, Hieu T e Arnold Smeulders: *Active learning using pre-clustering*. Em *Proceedings of the Twenty-First International Conference on Machine Learning*, página 79. Association for Computing Machinery, 2004. 27, 28

[105] Tong, Simon: *Active learning: theory and applications*. Tese de Doutoramento, Stanford University USA, 2001. 28

[106] Hinton, Geoffrey E. e Drew Van Camp: *Keeping the neural networks simple by minimizing the description length of the weights*. Em *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, páginas 5–13, 1993. 29

[107] Barber, David e Christopher M Bishop: *Ensemble learning in bayesian neural networks*. Nato ASI Series F Computer and Systems Sciences, 168:215–238, 1998. 29

[108] Blundell, Charles, Julien Cornebise, Koray Kavukcuoglu e Daan Wierstra: *Weight uncertainty in neural network*. Em *International Conference on Machine Learning*, páginas 1613–1622. PMLR, 2015. 29

[109] Gal, Yarin e Zoubin Ghahramani: *Bayesian convolutional neural networks with bernoulli approximate variational inference*. arXiv preprint arXiv:1506.02158, 2015. 30, 31

[110] Srivastava, Nitish, Geoffrey E. Hinton, Alexander Krizhevsky, Ilya Sutskever e Ruslan Salakhutdinov: *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. Journal of Machine Learning Research, 15, June 2014. 30

[111] Gal, Yarin e Zoubin Ghahramani: *Dropout as a bayesian approximation: Insights and applications*. Em *Deep Learning Workshop, ICML*, volume 1, página 2, 2015. 30

[112] Zeng, Jiaming, Adam Lesnikowski e Jose M Alvarez: *The relevance of bayesian layer positioning to model uncertainty in deep bayesian active learning*. arXiv preprint arXiv:1811.12535, 2018. 31

[113] Rokach, Lior: *Ensemble-based classifiers*. Artificial Intelligence Review, 33(1-2):1–39, 2010. 31

[114] Hansen, Lars Kai e Peter Salamon: *Neural network ensembles*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(10):993–1001, 1990. 32

[115] Lakshminarayanan, Balaji, Alexander Pritzel e Charles Blundell: *Simple and scalable predictive uncertainty estimation using deep ensembles*. arXiv preprint arXiv:1612.01474, 2016. 32

[116] Pawlowski, Nick, Miguel Jaques e Ben Glocker: *Efficient variational bayesian neural network ensembles for outlier detection*. arXiv preprint arXiv:1703.06749, 2017. 32

[117] Beluch, William H, Tim Genewein, Andreas Nürnberger e Jan M Köhler: *The power of ensembles for active learning in image classification*. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 9368–9377, 2018. 32, 33, 35, 36, 61, 62, 75, 104

[118] Gal, Yarin, Riashat Islam e Zoubin Ghahramani: *Deep bayesian active learning with image data*. Em *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, páginas 1183–1192. JMLR. org, 2017. 33, 34, 35, 37, 61, 62, 74, 104

[119] Zhu, Xiaojin, John Lafferty e Zoubin Ghahramani: *Combining active learning and semi-supervised learning using gaussian fields and harmonic functions*. Em *ICML 2003 workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, volume 3, páginas 58 – 65, 2003. 35

[120] Krizhevsky, Alex e Hinton: *Learning multiple layers of features from tiny images*. Relatório Técnico, 2009. 35

[121] Huang, Gao, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft e Kilian Q Weinberger: *Snapshot ensembles: Train 1, get m for free*. arXiv preprint arXiv:1704.00109, 2017. 35

[122] Yang, Hao, Chunfeng Yuan, Junliang Xing e Weiming Hu: *Diversity encouraging ensemble of convolutional networks for high performance action recognition*. Em *2017 IEEE International Conference on Image Processing (ICIP)*, páginas 2846–2850. IEEE, 2017. 35

[123] Osband, Ian, Charles Blundell, Alexander Pritzel e Benjamin Van Roy: *Deep exploration via bootstrapped dqn*. Em *Proceedings of the 30th International Conference on Neural Information Processing Systems*, páginas 4033–4041, 2016. 35

[124] Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens e Zbigniew Wojna: *Rethinking the inception architecture for computer vision*. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 2818–2826, 2016. 36

[125] Wang, Keze, Dongyu Zhang, Ya Li, Ruimao Zhang e Liang Lin: *Cost-effective active learning for deep image classification*. IEEE Transactions on Circuits and Systems for Video Technology, 27(12):2591–2600, 2017. 37, 38

[126] Shen, Yeji, Yuhang Song, Hanhan Li, Shahab Kamali, Bin Wang e C C Jay Kuo: *K-covers for active learning in image classification*. Em *2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, páginas 288–293. IEEE, 2019. 38

[127] Wang, Yu Emma, Gu Yeon Wei e David Brooks: *Benchmarking tpu, gpu, and cpu platforms for deep learning*. arXiv preprint arXiv:1907.10701, 2019. 40

[128] Cheng, Yu, Duo Wang, Pan Zhou e Tao Zhang: *A survey of model compression and acceleration for deep neural networks*. arXiv preprint arXiv:1710.09282, 2017. 40, 65

[129] Xu, Sheng, Anran Huang, Lei Chen e Baochang Zhang: *Convolutional neural network pruning: A survey*. Em *2020 39th Chinese Control Conference (CCC)*, páginas 7458–7463. IEEE, 2020. 40, 64

[130] Radu, Valentin, Kuba Kaszyk, Yuan Wen, Jack Turner, José Cano, Elliot J Crowley, Björn Franke, Amos Storkey e Michael O'Boyle: *Performance aware convolutional neural network channel pruning for embedded gpus*. Em *2019 IEEE International Symposium on Workload Characterization (IISWC)*, páginas 24–34. IEEE, 2019. 40

[131] Wu, Yawen, Zhepeng Wang, Yiyu Shi e Jingtong Hu: *Enabling on-device cnn training by self-supervised instance filtering and error map pruning*. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 39(11):3445–3457, 2020. 40, 64

[132] Lemaire, Carl, Andrew Achkar e Pierre Marc Jodoin: *Structured pruning of neural networks with budget-aware regularization*. Em *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, páginas 9108–9116, 2019. 40

[133] Li, Yuchao, Shaohui Lin, Baochang Zhang, Jianzhuang Liu, David Doermann, Yongjian Wu, Feiyue Huang e Rongrong Ji: *Exploiting kernel sparsity and entropy for interpretable cnn compression.* Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 2800–2809, 2019. 40

[134] Zou, Junhua, Ting Rui, You Zhou, Chengsong Yang e Sai Zhang: *Convolutional neural network simplification via feature map pruning.* Computers & Electrical Engineering, 70:950–958, 2018. 40, 41, 45, 65, 66

[135] Luo, Jian Hao, Hao Zhang, Hong Yu Zhou, Chen Wei Xie, Jianxin Wu e Weiyao Lin: *Thinet: pruning cnn filters for a thinner net.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(10):2525–2538, 2018. 41, 42, 44, 64, 65

[136] Osaku, Daniel, JF Gomes e Alexandre X Falcão: *Convolutional neural network simplification with progressive retraining.* arXiv preprint arXiv:2101.04699, 2021. 42, 65

[137] Jin, Haifeng, Qingquan Song e Xia Hu: *Auto-keras: An efficient neural architecture search system.* Em *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, páginas 1946–1956, 2019. 42

[138] Cai, Han, Tianyao Chen, Weinan Zhang, Yong Yu e Jun Wang: *Efficient architecture search by network transformation.* Em *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 42

[139] He, Xin, Kaiyong Zhao e Xiaowen Chu: *Automl: A survey of the state-of-the-art.* Knowledge-Based Systems, 212:106622, 2021. 42

[140] Lin, Shaohui, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang e David Doermann: *Towards optimal structured cnn pruning via generative adversarial learning.* Em *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, páginas 2790–2799, 2019. 42, 44, 64, 65

[141] Ding, Xiaohan, Tianxiang Hao, Jianchao Tan, Ji Liu, Jungong Han, Yuchen Guo e Guiguang Ding: *Resrep: Lossless cnn pruning via decoupling remembering and forgetting.* Em *Proceedings of the IEEE/CVF International Conference on Computer Vision and Pattern Recognition*, páginas 4510–4520, 2021. 42, 43, 64, 65, 66

[142] He, Kaiming, Xiangyu Zhang, Shaoqing Ren e Jian Sun: *Identity mappings in deep residual networks.* Em *European Conference on Computer Vision*, páginas 630–645. Springer, 2016. 44, 45, 69, 89, 96

[143] Saltz, Joel, Rajarsi Gupta, Le Hou, Tahsin Kurc, Pankaj Singh, Vu Nguyen, Dimitris Samaras, Kenneth R Shroyer, Tianhao Zhao, Rebecca Batiste *et al.*: *Spatial organization and molecular correlation of tumor-infiltrating lymphocytes using deep learning on pathology images.* Cell Reports, 23(1):181, 2018. 46, 47, 49, 50, 72

[144] Abousamra, Shahira, Le Hou, Rajarsi Gupta, Chao Chen, Dimitris Samaras, Tahsin Kurc, Rebecca Batiste, Tianhao Zhao, Shroyer Kenneth e Joel Saltz: *Learning from thresholds: Fully automated classification of tumor infiltrating lymphocytes for multiple cancer types.* arXiv preprint arXiv:1907.03960, 2019. 46, 47

[145] Oble, Darryl A, Robert Loewe, Ping Yu e Martin C Mihm: *Focus on tils: prognostic significance of tumor infiltrating lymphocytes in human melanoma.* Cancer Immunity Archive, 9(1), 2009. 46

[146] Angell, Helen e Jérôme Galon: *From the immune contexture to the immunoscore: the role of prognostic and predictive immune markers in cancer.* Current Opinion in Immunology, 25(2):261–267, 2013. 46

[147] Mlecnik, Bernhard, Gabriela Bindea, Franck Pagès e Jérôme Galon: *Tumor immunosurveillance in human cancers.* Cancer Metastasis Reviews, 30(1):5–12, março 2011. 46

[148] Teng, Michele WL, Shin Foong Ngiow, Antoni Ribas e Mark J Smyth: *Classifying cancers based on t-cell infiltration and pd-l1.* Cancer Research, 75(11):2139–2145, 2015. 46

[149] Han, S, C Zhang, Q Li, J Dong, Y Liu, Y Huang, T Jiang e A Wu: *Tumour-infiltrating cd4+ and cd8+ lymphocytes as predictors of clinical outcome in glioma.* British Journal of Cancer, 110(10):2560, 2014. 46

[150] Gurcan, Metin N, Laura E Boucheron, Ali Can, Anant Madabhushi, Nasir M Rajpoot e Bulent Yener: *Histopathological image analysis: A review.* IEEE Reviews in Biomedical Engineering, 2:147–171, 2009. 47

[151] Madabhushi, Anant e George Lee: *Image analysis and machine learning in digital pathology: Challenges and opportunities.* Medical Image Analysis, 33:170–175, 2016. 47

[152] Li, Chen, Dan Xue, Zhijie Hu, Hao Chen, Yudong Yao, Yong Zhang, Mo Li, Qian Wang e Ning Xu: *A survey for breast histopathology image analysis using classical and deep neural networks.* Em *International Conference on Information Technologies in Biomedicine*, páginas 222–233. Springer, 2019. 47

[153] Komura, Daisuke e Shumpei Ishikawa: *Machine learning methods for histopathological image analysis.* Computational and Structural Biotechnology Journal, 16:34–42, 2018. 47

[154] Wang, Shidan, Donghan M Yang, Ruichen Rong, Xiaowei Zhan e Guanghua Xiao: *Pathology image analysis using segmentation deep learning algorithms.* The American journal of pathology, 2019. 47

[155] Wang, Shidan, Donghan M Yang, Ruichen Rong, Xiaowei Zhan, Junya Fujimoto, Hongyu Liu, John Minna, Ignacio Ivan Wistuba, Yang Xie e Guanghua Xiao: *Artificial intelligence in lung cancer pathology image analysis.* Cancers (Basel), 11(11):1673, October 2019. 47

[156] Linder, Nina, Jenny C Taylor, Richard Colling, Robert Pell, Edward Alveyn, Johnson Joseph, Andrew Protheroe, Mikael Lundin, Johan Lundin e Clare Verrill: *Deep learning for detecting tumour-infiltrating lymphocytes in testicular germ cell tumours.* Journal of Clinical Pathology, 72(2):157–164, 2019. 47

[157] Klauschen, Frederick, K R Müller, Alexander Binder, Michael Bockmayr, M Hägele, P Seegerer, Stephan Wienert, Giancarlo Pruneri, S de Maria, S Badve *et al.*: *Scoring of tumor-infiltrating lymphocytes: From visual estimation to machine learning.* Em *Seminars in Cancer Biology*, volume 52, páginas 151–157. Elsevier, October 2018. 47

[158] Garcia, Emilio, Renato Hermoza, Cesar Beltran Castanon, Luis Cano, Miluska Castillo e Carlos Castanneda: *Automatic lymphocyte detection on gastric cancer ihc images using deep learning.* Em *2017 IEEE 30th international symposium on computer-based medical systems (CBMS)*, páginas 200–204. IEEE, 2017. 47

[159] Amgad, Mohamed, Elisabeth Specht Stovgaard, Eva Balslev, Jeppe Thagaard, Weijie Chen, Sarah Dudgeon, Ashish Sharma, Jennifer K Kerner, Carsten Denkert, Yinyin Yuan *et al.*: *Report on computational assessment of tumor infiltrating lymphocytes from the international immuno-oncology biomarker working group.* NPJ breast cancer, 6(1):1–13, 2020. 47

[160] Roy, Kaushiki, Debapriya Banik, Debotosh Bhattacharjee e Mita Nasipuri: *Patch-based system for classification of breast histology images using deep learning.* Computerized Medical Imaging and Graphics, 71:90–103, 2019. 47

[161] Berghoff, Anna S, Elisabeth Fuchs, Gerda Ricken, Bernhard Mlecnik, Gabriela Bindea, Thomas Spanberger, Monika Hackl, Georg Widhalm, Karin Dieckmann, Daniela Prayer, Amelie Bilocq, Harald Heinzl, Christoph Zielinski, Rupert Bartsch, Peter Birner, Jerome Galon e Matthias Preusser: *Density of tumor-infiltrating lymphocytes correlates with extent of brain edema and overall survival time in patients with brain metastases.* OncoImmunology, 5(1):e1057388, 2016. `https://doi.org/10.1080/2162402X.2015.1057388`, PMID: 26942067. 51

[162] Dahlin, Anna M, Maria L Henriksson, Bethany Van Guelpen, Roger Stenling, Åke Öberg, Jörgen Rutegård e Richard Palmqvist: *Colorectal cancer prognosis depends on t-cell infiltration and molecular characteristics of the tumor.* Modern Pathology, 24(5):671–682, 2011. `https://doi.org/10.1038/modpathol.2010.234`. 51

[163] Le, Han, Rajarsi R. Gupta, Le Hou, Shahira Abousamra, Danielle Fassler, Tahsin M. Kurç, Dimitris Samaras, Rebecca C. Batiste, Tianhao Zhao, Alison L. Van Dyke, Ashish Sharma, Erich Bremer, Jonas S. Almeida e J. Saltz: *Utilizing Automated Breast Cancer Detection to Identify Spatial Distributions of Tumor Infiltrating Lymphocytes in Invasive Breast Cancer.* The American Journal of Pathology, 190, 2020. 51, 70, 85

[164] Sculley, David: *Web-Scale k-Means Clustering.* Em *Proceedings of the 19th International Conference on World Wide Web*, páginas 1177–1178. Association for Computing Machinery, 2010. 55

[165] Miao, Runtian, Robert Toth, Yu Zhou, Anant Madabhushi e Andrew Janowczyk: *Quick annotator: an open-source digital pathology based rapid image annotation tool.* The Journal of Pathology. Clinical Research, 7(6):542–547, November 2021. 57, 105

[166] Queipo, Nestor V., Raphael T. Haftka, Wei Shyy, Tushar Goel, Rajkumar Vaidyanathan e P. Kevin Tucker: *Surrogate-based analysis and optimization*. Progress in Aerospace Sciences, 41(1):1–28, 2005, ISSN 0376-0421. https://www.sciencedirect.com/science/article/pii/S0376042105000102. 62

[167] Gomes, Jeremias, Willian Barreiros, Tahsin Kurc, Alba C.M.A. Melo, Jun Kong, Joel H. Saltz e George Teodoro: *Sensitivity analysis in digital pathology: Handling large number of parameters with compute expensive workflows*. Computers in Biology and Medicine, 108:371–381, 2019, ISSN 0010-4825. https://www.sciencedirect.com/science/article/pii/S0010482519300782. 62

[168] Bianco, Simone, Remi Cadene, Luigi Celona e Paolo Napoletano: *Benchmark analysis of representative deep neural network architectures*. IEEE Access, 6:64270–64277, 2018. 63

[169] Ye, Xucheng, Pengcheng Dai, Junyu Luo, Xin Guo, Yingjie Qi, Jianlei Yang e Yiran Chen: *Accelerating cnn training by pruning activation gradients*. Em *European Conference on Computer Vision*, páginas 322–338. Springer, 2020. 64

[170] *What's the backward-forward FLOP ratio for Neural Networks?*, December 2021. https://www.lesswrong.com/posts/fnjKpBoWJXcSDwhZk/what-s-the-backward-forward-flop-ratio-for-neural-networks. 64

[171] Han, Song, Huizi Mao e William J Dally: *Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding*. arXiv preprint arXiv:1510.00149, 2015. 64

[172] Lin, Shaohui, Rongrong Ji, Yuchao Li, Yongjian Wu, Feiyue Huang e Baochang Zhang: *Accelerating convolutional networks via global & dynamic filter pruning.* Em *IJCAI*, volume 2, página 8, 2018. 64

[173] Shao, Mingwen, Junhui Dai, Jiandong Kuang e Deyu Meng: *A dynamic cnn pruning method based on matrix similarity*. Signal, Image and Video Processing, 15(2):381–389, 2021. 64, 65

[174] Ding, Xiaohan, Xiangxin Zhou, Yuchen Guo, Jungong Han, Ji Liu *et al.*: *Global sparse momentum sgd for pruning very deep neural networks*. Advances in Neural Information Processing Systems, 32, 2019. 64

[175] Han, Song, Jeff Pool, John Tran e William Dally: *Learning both weights and connections for efficient neural network*. Advances in Neural Information Processing Systems, 28, 2015. 64

[176] Ba, Jimmy e Rich Caruana: *Do deep nets really need to be deep?* Advances in neural information processing systems, 27, 2014. 64

[177] Banner, Ron, Yury Nahshan e Daniel Soudry: *Post training 4-bit quantization of convolutional networks for rapid-deployment*. Advances in Neural Information Processing Systems, 32, 2019. 64

[178] He, Yang, Ping Liu, Ziwei Wang, Zhilan Hu e Yi Yang: *Filter pruning via geometric median for deep convolutional neural networks acceleration.* Em *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, páginas 4340–4349, 2019. 64

[179] Ding, Xiaohan, Guiguang Ding, Yuchen Guo, Jungong Han e Chenggang Yan: *Approximated oracle filter pruning for destructive cnn width optimization.* Em *International Conference on Machine Learning*, páginas 1607–1616. PMLR, 2019. 65

[180] Lu, Zhou, Hongming Pu, Feicheng Wang, Zhiqiang Hu e Liwei Wang: *The expressive power of neural networks: A view from the width.* Em *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, páginas 6232–6240, Red Hook, NY, USA, 2017. Curran Associates Inc., ISBN 9781510860964. 66

[181] Hajabdollahi, Mohsen, Reza Esfandiarpoor, Kayvan Najarian, Nader Karimi, Shadrokh Samavi e SM Reza Soroushmehr: *Hierarchical pruning for simplification of convolutional neural networks in diabetic retinopathy classification.* Em *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, páginas 970–973. IEEE, 2019. 66

[182] Tan, Mingxing e Quoc V Le: *Efficientnet: Rethinking model scaling for convolutional neural networks.* arXiv preprint arXiv:1905.11946, 2019. 66, 68

[183] Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke e Alexander A Alemi: *Inception-v4, inception-resnet and the impact of residual connections on learning.* Em *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 70, 76, 89, 99

[184] National Human Genome Research Institute: *The Cancer Genome Atlas.* https://cancergenome.nih.gov/, June 2017. `https://cancergenome.nih.gov/`, acesso em 2017 June 14. 72

[185] Chollet, François: *Simple MNIST convnet*, julho 2020. `https://keras.io/examples/vision/mnist_convnet/`. 74

[186] Jung, Alexander: *Imgaug*, July 2022. `https://imgaug.readthedocs.io/en/latest/`. 85

[187] Chollet, François: *Xception: Deep learning with depthwise separable convolutions.* Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 1800–1807. IEEE, 2017. 96

[188] Fan, Jianqing e Runze Li: *Statistical challenges with high dimensionality: Feature selection in knowledge discovery.* arXiv preprint math/0602133, 2006. 99

[189] Donahue, Jeff, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng e Trevor Darrell: *Decaf: A deep convolutional activation feature for generic visual recognition.* arXiv:1310.1531 [cs.CV], October 2013. 100

[190] Kattenborn, Teja, Jens Leitloff, Felix Schiefer e Stefan Hinz: *Review on convolutional neural networks (cnn) in vegetation remote sensing.* ISPRS Journal of Photogrammetry and Remote Sensing, 173:24–49, 2021. 105