
Universidade Estadual de Feira De Santana

Programa de Pós-Graduação em Computação Aplicada

# Detecção Automática de Glomérulos em Imagens Histológicas Renais Digitais

Jonathan Moreira Cardozo Rehem

Feira de Santana

2019


Universidade Estadual de Feira De Santana

Programa de Pós-Graduação em Computação Aplicada

Jonathan Moreira Cardozo Rehem

# Detecção Automática de Glomérulos em Imagens Histológicas Renais Digitais

Dissertação apresentada à Universidade Estadual de Feira de Santana como parte dos requisitos para a obtenção do título de Mestre em Computação Aplicada.

Orientador: Profa. Dra. Michele Fúlvia Angelo

Coorientador: Dr. Washington Luís Conrado dos Santos

Feira de Santana

2019

**Ficha catalográfica - Biblioteca Central Julieta Carteado - UEFS**

R27d Rehem, Jonathan Moreira Cardozo
Detecção automática de glomérulos em imagens histológicas renais digitais / Jonathan Moreira Cardozo Rehem. - 2019.
95 f. : il.

Orientadora: Michele Fúlvia Angelo.
Coorientador: Washington Luís Conrado dos Santos.

Dissertação (mestrado) - Universidade Estadual de Feira de Santana, Programa de Pós-Graduação em Computação Aplicada, 2019.

1. Rim - Processamento de imagem assistida por computador. 2. Rim - Doenças - Diagnóstico por imagem. I. Anelo, Michele Fúlvia, orient. II. Santos, Washington Luís Conrado dos, coorient. III. Universidade Estadual de Feira de Santana. III. Título.

CDU: 611.61:004.932

Clemilda Santana dos Reis de Jesus – Bibliotecária CRB5/1641

Jonathan Moreira Cardozo Rehem

**Detecção Automática de Glomérulos em Imagens Histológicas Renais Digitais**

Dissertação apresentada à Universidade Estadual de Feira de Santana como parte dos requisitos para a obtenção do título de Mestre em Computação Aplicada.

Feira de Santana, 20 de agosto de 2019

**BANCA EXAMINADORA**

Dra. Michele Fúlvia Angelo (Orientador)
Universidade Estadual de Feira de Santana

Dr. Angelo Amâncio Duarte
Universidade Estadual de Feira de Santana

Dr. Daniel Abensur Athanazio
Universidade Federal da Bahia

# Abstract

*Glomerulopathies, kidney diseases, affect thousands of people in Brazil and in the entire world, this number is growing constantly. The glomeruli are microscopic structures present in kidney and your examination by a doctor determines the kind and the degree of kidney disease. Kidney tissue images can be scanned or photographed, enabling the computational processing. Nowadays, detection and segmentation are made manually by a pathologist doctor. Thus, this research aims at propose a glomeruli automatic detection method on histological digital kidney tissue images. For this, we use deep learning techniques to train capable models to automate this task. Digital images photographed in varied approximation scales was used to compose train and test datasets. Tensorflow Object Detection API (Application Programming Interface) framework was used implements, train and test the models SSD (Single Shot Detection) Inception V2(SI2) and Faster RCNN (Region-based Convolutional Neural Network) Inception V2 (FRI2). Reaching 0.8831 mAP - 0.94 F1 Score when using the SI2 model, 0.8723 mAP and 0.97 F1 Score when utilizing FRI2 model. The SI2 model. The SI2 model is the most efficient for this task because it is 64% faster in training time and 98% faster in detecting glomeruli in each image. This work demonstrate the efficiency of deep learning techniques as solution for this problem, advancing the improvement of techniques for gloeruli automated detection.*

***Keywords:*** *Deep learning, Convolutional Neural Networks, Glomeruli, Object Detection.*

# Resumo

As glomerulopatias, doenças renais, acometem milhares de pessoas no Brasil e no mundo e este número vem crescendo. Os glomérulos são estruturas microscópicas presentes nos rins e sua análise por um médico patologista é o que determina o tipo e o grau da doença renal. Imagens dos tecidos renais podem ser digitalizadas ou fotografadas, o que torna possível o processamento por computador. Atualmente, a detecção e a separação de glomérulos é feita manualmente pelo patologista. Assim, esta pesquisa tem como objetivo propor um método de detecção automático de glomérulos em imagens histológicas renais digitais. Para isso, foram utilizadas técnicas de aprendizagem profunda a fim de treinar modelos que fossem capazes de automatizar esta tarefa. Imagens digitais de lâminas histológicas fotografadas em variadas escalas de aproximação foram utilizadas para compor os *datasets* de treinamento e testes. O *framework Tensorflow Object Detection API* foi utilizado como plataforma de implementação no treinamento e testes dos modelos SSD Inception V2 e Faster RCNN Inception V2. Obteve-se 0.8831 mAP e 0.94 *F1 Score* utilizando o modelo SI2, e 0.8723 mAP e 0.97 *F1 Score* utilizando o modelo FRI2. O modelo SI2 é o mais eficiente para esta tarefa, já que é 64% mais rápido no tempo necessário para o treinamento e 98% mais rápido na detecção de glomérulos em cada imagem. Este trabalho demonstra a eficiência do *Deep Learning* na resolução deste problema, avançando no aperfeiçoamento das técnicas de detecção automática de glomérulos.

**Palavras-chave:** *Deep learning*, Redes Neurais Convolucionais, Glomérulos, Detecção de Objetos.

# Prefácio

Esta dissertação de mestrado foi submetida a Universidade Estadual de Feira de Santana (UEFS) como requisito parcial para obtenção do grau de Mestre em Computação Aplicada.

A dissertação foi desenvolvida dentro do Programa de Pós-Graduação em Computação Aplicada (PGCA) tendo como orientadora a **Profa. Dra. Michele Fúlvia Angelo** e co-orientador o Dr. **Washington Luís Conrado dos Santos**.

# Agradecimentos

Agradeço a meus pais, dona Linda e seu Joaquim, por terem me dado a vida. Por terem depositado em mim todo amor e cuidado, tornando possível meu desenvolvimento com saúde, vontade aprender e de ajudar o próximo. Agradeço todos os sacrifícios que tiveram de fazer, sei que foram muitos.

Agradeço a meu falecido avô de criação, Sr. Antônio Andrade Campos, que por amor me criou como um filho desde os meus dois anos de idade. Me deu amor e afeto, investiu em minha educação, me ensinou como ser um homem honrado, me ensinou a perseguir meus sonhos, mesmo os que ele não entendia. Gostaria de ter tido tempo de te mostrar os frutos da sua colheita. Obrigado vô.

Agradeço a minha esposa Ariane por estar sempre ao meu lado, você é pessoa responsável por tudo que sou e por tudo que somos, obrigado pelo incentivo de cada dia e por entender minha ausência necessária as vezes, você é tudo pra mim. Muito obrigado.

Agradeço a minha sogra Gildé e meu sogro Edivaldo, obrigado pelo apoio constante.

Agradeço a professora Michele por toda dedicação, empenho, atenção e disponibilidade dedicados a mim e a este trabalho, sua ajuda foi fundamental.

Agradeço também aos professoras do PGCA, à UEFS e ao Estado da Bahia pela educação de qualidade a mim ofertados.

Agradeço ainda ao Dr. Washington e a FIOCRUZ pelas imagens fornecidas e as orientações prestadas.

# Sumário

# Lista de Tabelas

# Lista de Figuras

# Lista de Abreviações

| **Abreviação** | **Descrição** |
|---|---|
| API | *Application Programming Interface* |
| AR | *Average Recall* |
| CAD | *Computer Assisted Diagnosis* |
| CNN | *Convolutional Neural Networks* |
| COCO | *Common Objects in Context* |
| ConvNets | *Convolutional Networks* |
| CpqGM | Centro de Pesquisa Gonçalo Muniz |
| CPU | Computer Processing Unit |
| CR | Congo Red |
| DCDP | *Devide and Conquer Dynamic Program* |
| EDP | *Exhaustiv Dynamic Program* |
| ET-FL | *Extremely Randomized Trees for Feature Learning* |
| FC | *Fully Connected* |
| FIOCRUZ | Fundação Oswaldo Cruz |
| GIF | *Graphics Interchange Format* |
| GPU | *Graphics Processing Units* |
| H&E | Hematoxilina e Eosina |
| HOG | *Histogram of Oriented Gradients* |
| ILSVRC | *ImageNet Large Scale Visual Recognition Competition* |
| IoU | *Intersection Over Union* |
| JPEG | *Joint Photographic Experts Groups* |
| LBP | *Local Binary Pattern* |
| mAP | *Mean Average Precision* |
| MrcLBP | *Multi Radial Local Binary Pattern* |
| PAS | *Periodic Acid Schiff* |
| PASCAL VOC | *Pascal Visual Object Classes* |
| PGCA | Pós-Graduação em Computação Aplicada |
| R-CNN | *Region-Based Convolutional Neural Networks* |
| R-HOG | *Rectangular Histogram of Oriented Gradients* |
| RAM | Random Access Memory |
| ReLU | *Rectified Linear Units* |
| ResNet | *Residual Networks* |
| RGB | *Red, Green and Blue* |
| RMSProp | *Root Mean Square Propagation* |
| RoI | *Region of Interest* |

| | |
|---|---|
| S-HOG | *Segmental Histogram of Oriented Gradients* |
| SBN | Sociedade Brasileira de Nefrologia |
| SSD | *Single Shot Detection* |
| SSD | *Solid State Drive* |
| SVM | *Support Vector Machine* |
| TIFF | *Tagged Image File Format* |
| TOD | *Tensorflow Object Detection API* |
| UEFS | Universidade Estadual de Feira de Santana |
| USRDS | *United States Renal Data System* |
| vCPU | *Virtual Computer Processing Units* |
| WND-CHARM | *Multi-Purpose Image Classification Using Compound Transforms* |
| WSI | *Whole Slide Image* |
| XML | *Extensible Markup Language* |
| YOLO | *You Only Look Once* |

# **Capítulo 1** Introdução

As doenças renais atingem o organismo em sua capacidade de manter as condições normais de equilíbrio na composição sanguínea, livres de impurezas e toxinas, mantendo os componentes funcionais do sangue, a exemplo das hemácias, leucócitos e proteínas. Os órgãos responsáveis pela filtragem do sangue são os rins, parte importante dessa tarefa é realizada pelos glomérulos, que são microestruturas formadas por vasos capilares em formato de novelo de lã, localizados no néfron ou nefrônio, que são as unidades funcionais do rim. Cada rim é constituído por 1 a 4 milhões de néfrons (MELDAU, 2017).

As glomerulopatias são doenças renais que afetam os glomérulos. Devido a importância do papel dos glomérulos na filtragem sanguínea, a depender do tipo de lesão e da quantidade de glomérulos afetados, as glomerulopatias podem trazer consequências graves ao funcionamento do organismo, podendo levar o paciente a depender de tratamento permanente ou até mesmo levá-lo a morte.

Segundo Thomé et al. (2018), dados coletados pela Sociedade Brasileira de Nefrologia em 2017, publicado através do Censo Brasileiro de Diálise, em sua versão pública mais recente, conclui que o número de pacientes em tratamento dialítico continua a aumentar e a taxa de mortalidade elevou-se, foi estimado que em 2017 cerca de 126.583 pacientes estavam sob tratamento dialítico e taxa de mortalidade anual foi de 19,9%. Os tratamentos dialíticos tentam compensar a perda parcial ou total da capacidade renal na filtragem do sangue.

Segundo Saran et al. (2017), em dados coletados pelo USRDS (*US Renal Data System*) e publicados no relatório anual de 2016 pela *National Kidney Foundation*, nos Estados Unidos, foram reportados 120.688 novos casos de pacientes afetados por

doenças renais em estágio crônico no ano de 2014, totalizando 678.883 pacientes tratados em estágio crônico no ano de 2014. Este número continua a crescer, entre 2013 e 2014 nos Estados Unidos, houve um aumento de 1,1%. Foram gastos 32 bilhões de dólares pelo Sistema de Saúde Norte Americano no tratamento de pacientes renais crônicos no ano de 2014.

O diagnóstico das glomerulopatias é feito por critérios clínicos, exames laboratoriais e biópsia. Os exames laboratoriais de sangue podem indicar a presença de substâncias que normalmente são filtradas pelos rins. Exames laboratoriais da urina podem indicar a presença de substâncias que normalmente não são eliminadas pela urina, estes comportamentos evidenciam o mal funcionamento do sistema renal. Segundo Veronese et al. (2010), além dos exames laboratoriais, a análise histopatológica da biópsia renal também é necessária porque é através dela que o tipo de lesão glomerular é classificada e o planejamento prognóstico ou terapêutico do paciente é feito.

A biópsia renal consiste na extração de um fragmento de 1–2cm utilizando uma agulha específica ou por via cirúrgica, então esse fragmento de tecido é fixado quimicamente para evitar sua decomposição, cortado em secções de 2–3µm e preparados em lâminas que serão examinadas por um médico patologista utilizando um microscópio (KELLY; LANDMAN, 2014).

A histologia consiste no estudo microscópico de estruturas celulares e tecidos de organismos. De acordo com Belsare e Mushrif (2012), a histopatologia é o trabalho realizado pelo médico patologista com amostras de biópsia, utilizando um microscópio, o qual procura detectar a presença de diversos tipos de doenças, a exemplo do câncer. A análise histopatológica é um trabalho manual que requer muito tempo e atenção do médico patologista e é suscetível a variações intra e inter observador.

Com a evolução dos computadores tornou-se possível a digitalização de lâminas histológicas, convertendo-as para o formato digital, e assim podendo ser visualizadas

através de monitores e manipuladas utilizando-se ferramentas computacionais. Esse avanço deu origem a Histologia e Patologia Digital trazendo novas possibilidades de colaboração entre as áreas médica e da computação. Segundo Belsare e Mushrif (2012), o acesso às informações digitais de imagens histológicas possibilitou a análise auxiliada por computador utilizando algoritmos de processamento digital de imagens, de modo que o Diagnóstico Auxiliado por Computador (do inglês *Computer Assisted Diagnosis - CAD*) exercesse um papel muito importante, tornando-se o principal campo de pesquisa da histopatologia.

Segundo Sarder, Ginley e Tomaszewski (2016), a histopatologia digital traz consigo o potencial de reduzir drasticamente o tempo e o esforço manual requerido no diagnóstico das doenças renais. O auxílio computacional pode ajudar o médico patologista na automatização de parte de suas tarefas manuais, repetitivas e que consomem muito tempo. Podem também fornecer informações que o ajudem na tomada de decisão de forma mais segura e rápida, reduzindo a interferência inter e intra observador. O incremento da eficiência do diagnóstico abrevia o início do tratamento de doenças, que tem no tempo um fator importante nas chances da recuperação das funções renais dos pacientes.

Em uma revisão da literatura, observou-se que alguns trabalhos que tratam da análise de imagens histológicas digitais através do computador estão relacionados ao câncer de mama (WAN et al., 2016) (WAN et al., 2014), outros relacionadas ao câncer de próstata (MCCARTHY; CUNNINGHAM; OHURLEY, 2014). Poucos trabalhos estão voltados aos rins, recentemente, trabalhos voltados para a classificação automática de doenças renais ganharam importância, principalmente os relacionados ao sistema PathoSpotter (BARROS; DUARTE; SANTOS, 2015) (BARROS et al., 2017) (DE ARAÚJO et al., 2017).

Este trabalho é parte integrante dos esforços no desenvolvimento do PathoSpotter, um sistema que está sendo desenvolvido através de uma parceria entre especialistas da área médica e da área de computação para a classificação de lesões elementares a

partir de imagens histológicas renais e hepáticas (BARROS; DUARTE; SANTOS, 2015) (BARROS et al., 2017) (DE ARAÚJO et al., 2017). Entre os principais benefícios propostos pelo PathoSpotter, estão: o apoio à pesquisa sobre a relevância diagnóstica e prognóstica de lesões elementares glomerulares e o auxílio no treinamento de estudantes, médicos patologistas e clínicos. Destacamos que a capacidade de detectar glomérulos a partir de imagens histológicas é requisito para as tarefas de classificação que o PathoSpotter já faz, porém utiliza imagens de glomérulos recortadas manualmente. Com a nova funcionalidade proposta por este trabalho o sistema se tornará mais eficiente e robusto.

Através de uma revisão da literatura pode-se observar que pouco tem sido feito na pesquisa relacionada a detecção automática de glomérulos em imagens histológicas renais digitais de humanos. Foram encontrados trabalhos que tratam de detecção e segmentação, os trabalhos de Zhang, Hu e Zhu (2011) que não informaram de que espécie vieram as imagens; Kato et al. (2015) que utilizaram imagens histológicas de ratos; Marée et al. (2016) que utilizaram imagens histológicas de humanos; Sarder, Ginley, Tomaszewki (2016) que utilizaram imagens histológicas de ratos; Ginley et al. (2017) que utilizaram imagens de ratos e camundongos; Sarder, Ginley e Tomaszewski (2017) que utilizaram imagens de ratos e camundongos, Simon et al. (2018) que utilizaram imagens histológicas de ratos, camundongos e humanos e Gallego et al. (2018) que utilizaram imagens de humanos.

Assim, o objetivo deste trabalho é propor uma metodologia capaz de detectar automaticamente glomérulos em imagens histológicas digitais de humanos, a fim de aprimorar as capacidades do sistema PathoSpotter.

Esse trabalho auxiliará no aperfeiçoamento de algoritmos de classificação automática de lesões glomerulares, tarefa realizada pelo sistema PathoSpotter que atualmente é capaz de classificar glomerulopatias proliferativas com acurácia de 88.3 (BARROS; DUARTE; SANTOS, 2015) (BARROS et al., 2017) e glomerulosclerose segmentar com acurácia de 84.8 para imagens com corante H&E (Hematoxolina e Eosina) e 81.3

para imagens com corante PAS (*Periodic Acid Schiff*) (DE ARAÚJO et al., 2017), porém os classificadores automáticos atuam sobre imagens que contem somente um glomérulo por imagem.

Com a detecção automática dos glomérulos será possível também a alimentação automática de bancos de imagens histológicas de glomérulos, considerando que a diversidade dos bancos de imagens é um fator crítico para o treinamento de algoritmos de classificação automática, além de auxiliar no treinamento de novos médicos patologistas.

Portanto, pode-se verificar que os objetivos deste trabalho podem contribuir na eficiência do diagnóstico de doenças renais, no avanço científico das áreas relacionadas, no auxílio ao trabalho de médicos patologistas e no treinamento de novos médicos. Verifica-se também que, até o momento, a detecção automática de glomérulos em imagens histológicas humanas é um tema pouco explorado no meio científico, havendo chances de impactar positivamente.

# **Capítulo 2** Revisão Bibliográfica

Neste capítulo serão apresentados os conceitos necessários para o entendimento deste trabalho.

## 2.1 Conceitos Médicos

A seguir serão apresentados os termos médicos necessários para o entendimento deste trabalho.

### 2.1.1 Rins e Glomérulos

O rim é um órgão essencial para a manutenção da vida, além de ser responsável pela remoção de toxinas do sistema circulatório, também exerce outras funções como regulação na formação do sangue e dos ossos, regulação da pressão sanguínea, controle do delicado balanço químico e de líquidos do corpo. Os rins removem substâncias indesejadas que possam estar presentes na circulação sanguínea e as leva para fora do corpo através da urina, que é uma substância líquida formada nos rins e composta basicamente por água e dejetos. No corpo humano existe um par de rins, localizados abaixo do fígado e do baço, posicionados cada um de um lado da coluna vertebral (MELDAU, 2017) ("Glomerulopatias", 2017).

Na Figura 1 é apresentada uma ilustração das estruturas que formam o rim. Seu formato se assemelha a um grão de feijão, no centro do desenho à esquerda, estão suas conexões com veias e artérias numa zona chamada hilo renal. À direita destes estão as pirâmides, que em sua extremidade mais larga estão conectadas ao córtex renal. A região das pirâmides e do córtex constituem a parte funcional do rim, também chamada de parênquima ("Sistema Urinário", 2017) (MELDAU, 2017).

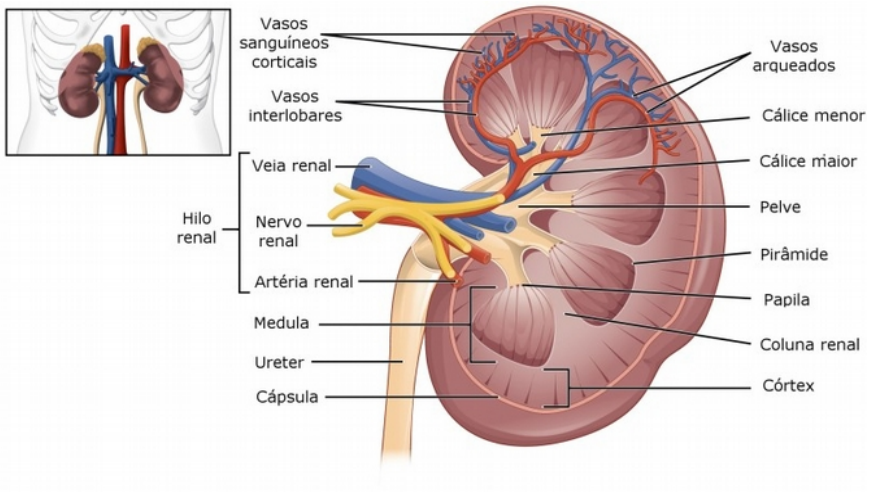



*Figura 1: Anatomia do Rim. Fonte: (MELDAU, 2017)*

No parênquima estão localizadas as unidades funcionais do rim, estruturas microscópicas chamadas néfrons ou nefrônios. Cada rim possui cerca de 1 milhão de néfrons. Ele é chamado de unidade funcional porque cada néfron é capaz de realizar todas as funções renais. Cada néfron é constituído pela cápsula de Bowman, pelo glomérulo e pelos túbulos renais. O glomérulo e a cápsula de Bowman formam uma estrutura denominada corpúsculo de Malpighi ("Sistema Urinário", 2017) (MELDAU, 2017).

Na Figura 2 é possível ver que o glomérulo é uma estrutura formada por vasos capilares em formato de novelo de lã, localizado dentro da capsula de Bowman. A arteríola aferente divide-se em várias alças capilares, formando o glomérulo. Esses capilares voltam a se unir formando a arteríola eferente que sai da cápsula de Bowman pelo polo vascular (MELDAU, 2017).

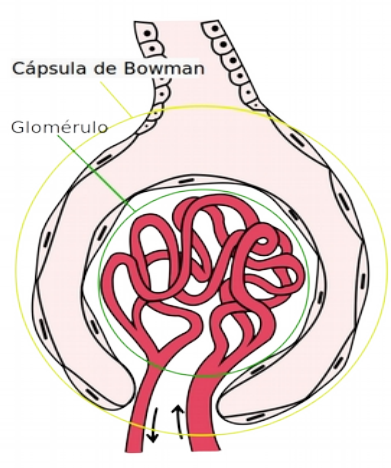



*Figura 2: Ilustração de um glomérulo.*
*Fonte: (GRAY, 1918) (adaptado).*

Quando o sangue passa pelos capilares glomerulares, água e outras substâncias saem do sangue, passam através das células endoteliais e caem no espaço de Bowman, de onde seguem para os túbulos renais. Esse líquido produzido pelo glomérulo recebe o nome de filtrado glomerular e o processo pelo qual ele se formou chama-se filtração glomerular. Nos túbulos renais o filtrado glomerular é processado e transformado em urina (MELDAU, 2017).

### 2.1.2 Glomerulopatias

As glomerulopatias, também conhecidas como glomerulonefrites, são doenças que acometem os glomérulos. “São doenças muito variadas, algumas de natureza aguda, outras de curso crônico; umas de caráter eminentemente inflamatório, outras não; algumas sabidamente tratáveis, outras não” (“Glomerulopatias”, 2017).

Pacientes acometidos com glomerulopatias podem não apresentar sintomas, ou apresentar sintomas urinários, como urina escura, diminuição do volume urinário, ou ainda inchaço dos membros inferiores, do rosto ou inchaço de todo o corpo (“Glomerulopatias”, 2017).

O exame de urina em pacientes acometidos por glomerulopatias pode revelar a presença de substâncias que normalmente não estão presentes na urina porque são filtrados pelos glomérulos. Quando há presença de hemácias na urina, esta condição

chama-se hematúria e quando há presença de proteínas, esta condição chama-se proteinúria (VERONESE et al., 2010).

Segundo Veronese et al. (2010), além dos exames laboratoriais, a análise histopatológica da biópsia renal também é necessária porque é através dela que o tipo de lesão glomerular é classificada e o planejamento prognóstico ou terapêutico do paciente é feito.

### 2.1.3 Histologia

Segundo Gartnet e Hiatt (2007), a histologia é um ramo da anatomia que estuda os tecidos dos animais e das plantas, mas não se limita somente ao estudo da estrutura do corpo, ela também trata das funções deste.

Para o estudo de tecidos, vasos, células e outras estruturas microscópicas, a histologia analisa amostras de tecido processadas e preparadas em lâminas de vidro, estudadas utilizando-se um microscópio. A histologia renal precisa extrair amostras do corpo do paciente e faz isso através da biópsia renal.

A biópsia renal é o procedimento em que, através de uma agulha específica ou por meio cirúrgico, um fragmento de 1–2cm é extraído do rim do paciente. Essa amostra então é preparada para a lâmina de vidro. Para isso, o fragmento é fixado com produtos químicos que preservam as características dos tecidos interrompendo a decomposição. Depois disso, são aplicados corantes que evidenciam certos aspectos da amostra baseado no tipo de estudo que será feito, é comum amostras diferentes da mesma região do tecido serem coradas com mais de um tipo de corante para analisar o tecido sob diferentes aspectos (JUNQUEIRA; CARNEIRO, 2013).

Na Figura 3, pode-se visualizar uma imagem histológica de um glomérulo saudável, vista através de microscópio e digitalizada posteriormente, o tecido foi corado utilizando corante do tipo PAS.

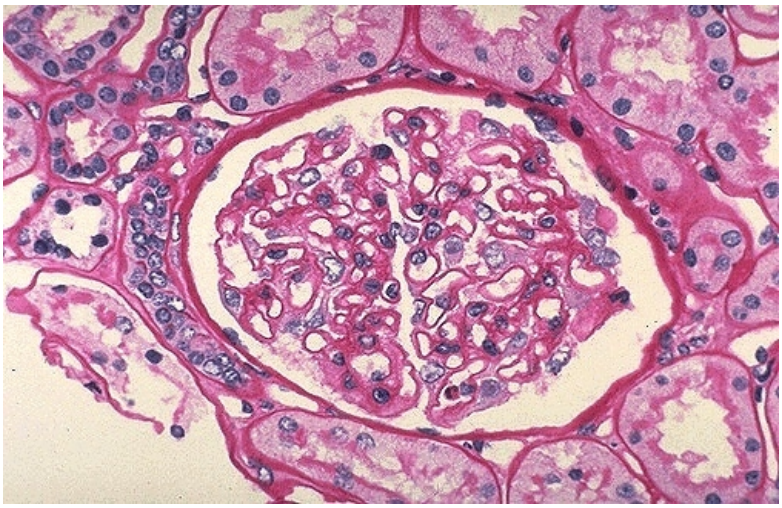


*Figura 3: Glomérulo saudável visto no microscópio. Fonte: (UNIVERSITY OF UTAH, 2017).*

### 2.1.4 Histopatologia Digital

De acordo com Belsare e Mushrif (2012), a histopatologia é o trabalho realizado pelo médico patologista com amostras de biópsia no microscópio, procurando detectar a presença de diversos tipos de doenças, a exemplo do câncer. A análise histopatológica é um trabalho manual que requer muito tempo e atenção do médico patologista e é suscetível a variações intra e inter observador.

Atualmente, com o advento dos *scanners* de lâmina histológica e dos microscópios digitais, é possível capturar toda a informação trazida pela lâmina em uma imagem digital, tornando possível sua manipulação através do computador. Essas capacidades criaram novas oportunidades para a histopatologia.

## 2.2 Conceitos da Computação

### 2.2.1 Visão Computacional

"Visão computacional pode ser definida como a área de estudo que tenta repassar para máquinas a incrível capacidade da visão." (BACKES; SÁ JUNIOR, 2016). O objetivo da visão computacional é desenvolver sistemas artificiais capazes de emular a visão natural biológica, por isso nesses sistemas a entrada de dados é uma imagem

digital e a saída é uma interpretação das informações contidas nessa imagem.

Uma imagem digital é uma representação digital de uma imagem bidimensional, ou tridimensional, que pode ter origem de diversas fontes, por exemplo, pode ser uma imagem capturada por dispositivos sensíveis à banda visual do espectro eletromagnético, os sensores fotográficos, ou podem ser capturadas por sensores capazes de captar sinais eletromagnéticos invisíveis ao olho humano como: raios-x, raios gama, infravermelho e ultrassom; ou ainda podem ser sintetizadas artificialmente a partir de equações matemáticas, como as imagens fractais e a modelagem 3D (GONZALEZ; WOODS, 2009).

Uma imagem digital é representada por uma matriz bidimensional e pode ser definida como "uma função bidimensional, $f\ (x,\ y)$, em que $x$ e $y$ são coordenadas espaciais (plano), e a amplitude de $f$ em qualquer par de coordenadas $(x,\ y)$ é chamada de intensidade ou nível de cinza da imagem nesse ponto. Quando $x$, $y$ e os valores de intensidade de $f$ são quantidades finitas e discretas, chamamos de imagem digital" (GONZALEZ; WOODS, 2009).

Segundo Gonzales e Woods (2009), não existe um limite claro na definição de Visão Computacional, mas é útil agrupar os processos relacionadas a ela em três níveis, as operações de baixo, médio e alto nível.

Operações de baixo nível são processos primitivos como o pré-processamento para redução de ruído na imagem, o realce de contraste e o aguçamento. Os processos de médio nível envolvem a segmentação (separação de regiões de imagens ou objetos) e a descrição do objeto a fim de reduzi-lo a uma forma adequada ao processamento computacional. Processos de alto nível tentam dar sentido a um conjunto de objetos reconhecidos nas imagens (GONZALEZ; WOODS, 2009).

### 2.2.2 Aprendizagem de Máquina

Segundo Hosch (2016), Aprendizagem de Máquina é um subtópico do ramo de Inteligência Artificial na Ciência da Computação, que estuda a implementação de

algoritmos capazes de aprender autonomamente. De acordo com Géron (2017), "Aprendizagem de Máquina é a ciência (e arte) de programar computadores de modo que possam aprender a partir de dados", ele diz ainda que a aprendizagem de máquina é indicada para problemas que são complexos demais para as abordagens tradicionais ou que se desconheça um algoritmo para tal problema. Esses algoritmos processam uma quantidade grande de dados em sua entrada, durante este processo tentam reconhecer padrões que destaquem características peculiares nos objetos com o objetivo de produzir um modelo que seja capaz de fazer previsões corretas quando novos dados forem fornecidos ao modelo.

Os algoritmos de aprendizagem de máquina podem processar os mais variados tipos de conjuntos de dados, desde dados numéricos simples, textos, sinais sonoros, até imagens e vídeos, desde que sejam preparados para atender ao tipo de entrada exigida pelos algoritmos. Cada algoritmo possui comportamentos peculiares que os tornam mais adequados a determinados fins e menos adequados a outros, o estudo dessas características e a experimentação são essenciais para alcançar os objetivos pretendidos.

Pode-se dizer que um modelo está sendo treinado quando este é exposto ao conjunto de dados que pretende-se processar a fim de aprender com suas características, os modelos podem ser classificados com base no modo em que aprendem. A aprendizagem supervisionada é o tipo mais comum e utiliza um conjunto de dados (*dataset*) etiquetado, ou seja, cada instância do conjunto acompanha a resposta correta para ela. Durante o treinamento, o algoritmo tenta associar padrões nos dados ao resultado esperado, ao final do processo de treinamento, o modelo deve ser capaz de deduzir qual seria a resposta correta para dados que não se conhece a resposta. Na aprendizagem não supervisionada, o *dataset* não é etiquetado, ou seja, ele não sabe qual é a resposta correta, o objetivo é que o modelo seja capaz de agrupar os dados em categorias considerando alguma característica de similaridade entre eles.

#### *2.2.2.1 Deep Learning*

*Deep Learning* ou Aprendizagem Profunda é um subcampo da Aprendizagem de Máquina (GOODFELLOW; BENGIO; COURVILLE, 2016) (CHOLLET, 2017), e é uma nova forma de alcançar a representação dos dados, pondo ênfase em gerar camadas sucessivas de representações cada vez mais significativas. O termo "profundo" aqui não traz o significado de alcançar um profundo entendimento do problema através dessa abordagem, mas sim, faz referência ao encadeamento sucessivo e cada vez mais abstrato das camadas que processam os dados de entrada (CHOLLET, 2017). Esses métodos melhoraram o estado da arte em reconhecimento de voz, reconhecimento visual de objetos, detecção de objetos e muitos outros domínios, como na descoberta de medicamentos e na genética (LECUN; BENGIO; HINTON, 2015). Na Figura 4 pode-se ver a posição ocupada pela *Deep Learning* no campo de Aprendizagem de Máquina e Inteligência Artificial.

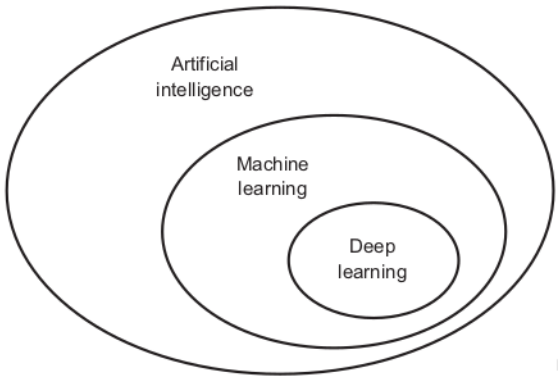



*Figura 4: Como se posiciona o Deep Learning (CHOLLET, 2017)*

Neste método, os algoritmos são chamados de RNA (Redes Neurais Artificiais), porque em sua concepção se basearam na anatomia humana, trazendo similaridades conceituais como neurônios artificiais que são unidades de processamento que compõe as arquiteturas e estão conectados em rede. Essas redes são organizadas em camadas (*layers*), os dados de entrada são fornecidos para a camada de entrada, que os processa fornecendo o resultado dessa operação como entrada para a próxima camada, esse processo se repete a depender da quantidades de camadas definidas pela arquitetura do algoritmo. Uma rede pode ter uma quantidade variável de camadas de

neurônios conectadas, a chamada profundidade (*deep*) da rede. Nenhum atributo é previamente extraído dos dados como é comum em outras técnicas de Aprendizagem de Máquina, os dados brutos são fornecidos e cada camada tenta aprender as características peculiares que possam ajudar a rede a mapear os dados de entrada com os resultados esperados na saída.

Cada camada executa transformações simples, mas não-lineares, nos dados que são parametrizados por pesos (*weights*), no início do processo de ajuste do modelo, etapa denominada treinamento. Os pesos são inicializados com valores randômicos, e a medida que o processo avança, os pesos são ajustados a fim de melhorar o desempenho. A técnica mais utilizada para esse fim é a B*ackpropagation*, onde ao final de cada etapa de treinamento, o gradiente da função de erro é calculado e utilizado para atualizar recursivamente os pesos usados pelo modelo, buscando minimizar o erro na próxima etapa de treinamento (Deep Learning Book, 2018) (CHOLLET, 2017).

Em tarefas de classificação em imagens, por exemplo, os dados são representados como um arranjo de pixels e o aprendizado nas primeiras camadas da rede tipicamente detectam a presença ou ausência de bordas em orientação e localização peculiares da imagem. A segunda camada tipicamente detecta arranjos particulares de bordas, mesmo havendo variações na posição das bordas. A terceira camada detecta arranjos maiores que a camada anterior que geralmente correspondem a partes do objeto, e as camadas subsequentes detectam objetos como a combinação desse arranjos. O aspecto chave do *Deep Learning* é que as camadas que identificam atributos não são projetadas por humanos, elas aprendem através dos dados, utilizando um procedimento de aprendizado de propósito geral (LECUN; BENGIO; HINTON, 2015).

Existem diversas arquiteturas de redes neurais artificiais, as ConvNets (*Convolutional Neural Networks* ou Redes Neurais Convolucionais) são fáceis de treinar e generalizar, conseguiram sucesso em atividades práticas e a adesão da comunidade de visão

computacional (LECUN; BENGIO; HINTON, 2015).

#### 2.2.2.1.1 Redes Neurais Convolucionais

As ConvNets, também conhecidas como CNN (do inglês *Convolutional Neural Networks*) são um tipo especial de Rede Neural Artificial que usa uma operação matemática chamada convolução. As redes convolucionais são simplesmente redes neurais artificiais que utilizam a convolução no lugar da multiplicação comum de matrizes em pelo menos uma de suas camadas (GOODFELLOW; BENGIO; COURVILLE, 2016). As ConvNets foram projetadas para processar dados que vem em forma de múltiplas matrizes, por exemplo, uma imagem colorida é composta por três matrizes 2D que contêm a intensidade de cor para cada pixel e para cada canal de cor. Muitos outros tipos de informação podem ser representados na forma de matrizes multidimensionais como, sinais de áudio, imagens 2D e 3D (LECUN; BENGIO; HINTON, 2015).

O compartilhamento de parâmetros sobre múltiplas posições na imagem torna as ConvNets invariantes à translação (deslocamento), ou seja, uma imagem que contém um gato continua sendo uma imagem contendo um gato mesmo que esse esteja deslocado um pixel para a direita, isso significa que podemos detectar gatos na imagem usando o mesmo detector, mesmo que este apareça na coluna $i$ ou na coluna $i + 1$ na imagem (GOODFELLOW; BENGIO; COURVILLE, 2016).

Existem aplicações de ConvNets desde o início da década de 1990 no reconhecimento de voz e leitura de documentos. Desde esse período existiram experimentos na detecção de objetos em imagens naturais como mãos, rostos e reconhecimento de faces. Desde os anos 2000 as ConvNets tem sido utilizadas e conseguido sucesso expressivo na detecção, segmentação e reconhecimento de objetos e regiões de imagens, em tarefas que os dados etiquetados eram relativamente abundantes como, reconhecimento de placas de trânsito, segmentação de imagens biológicas, detecção de faces e texto, pedestres e humanos em imagens naturais (LECUN; BENGIO; HINTON, 2015).

Apesar deste sucesso, as ConvNets foram completamente esquecidas até a competição ImageNet em 2012, quando foram aplicadas sobre um *dataset* de milhões de imagens adquiridas através da Internet, que continham 1.000 diferentes classes, eles alcançaram um resultado espetacular, que revolucionou a computação visual. Hoje, ConvNet é a abordagem mais empregada em quase todas as tarefas de reconhecimento e detecção de objetos, e se aproxima da capacidade humana em algumas tarefas (LECUN; BENGIO; HINTON, 2015).

A arquitetura típica de uma ConvNet está organizada em uma série de estágios. O primeiro estágio é composto por dois tipos de camadas: Camadas Convolucionais e Camadas de P*ooling*. O primeiro tipo, processa as imagens considerando campos receptivos locais, o segundo reduz a dimensionalidade espacial das representações. O próximo estágio é composto por uma camada totalmente conectada (*Fully Connected*) que atua como um classificador, determinando a probabilidade da imagem pertencer a uma determinada classe (PONTI; DA COSTA, 2018) (LECUN; BENGIO; HINTON, 2015).

De acordo com Ponti e Da Costa (2018) "A principal aplicação das CNNs é para o processamento de informações visuais, em particular imagens, pois a convolução permite filtrar as imagens considerando sua estrutura bidimensional (espacial)." Na Figura 5 é possível ver a arquitetura comum de uma ConvNet e suas camadas.

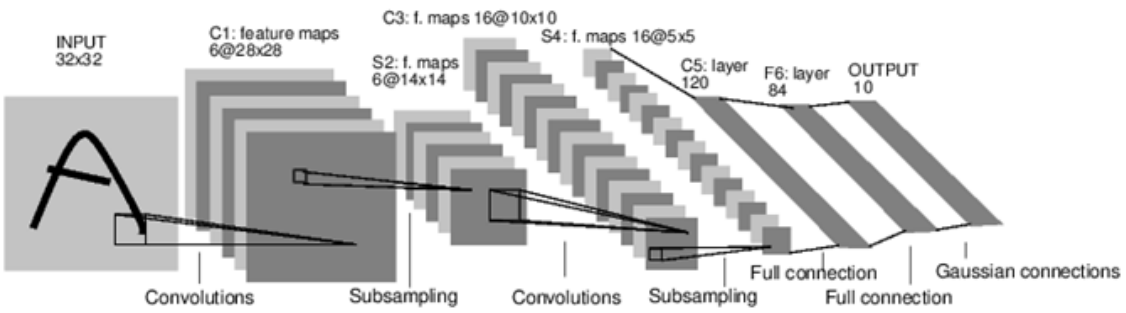



*Figura 5: A Arquitetura de uma ConvNet (LeNet). (DESHPANDE, 2016)*

#### 2.2.2.1.1.1 Camada Convolucional

A Camada Convolucional é o bloco fundamental das ConvNets e tem a função de detectar combinações de atributos na informação trazida pela camada anterior (LECUN; BENGIO; HINTON, 2015). Esta camada é a responsável por detectar atributos relevantes em uma imagem, ou volume, que tornam possível a distinção de objetos entre classes. Em uma rede ConvNet é comum existirem varias camadas convolucionais intercaladas com outros tipos, elas são conectadas à próxima camada de forma não linear. A primeira camada de uma rede ConvNet é sempre uma camada convolucional. Cada camada convolucional é formada por um conjunto de **filtros**, também chamados de máscaras, neurônios ou *kernel*, cada filtro é especializado em detectar formas. Nas primeiras camadas são detectadas formas simples como bordas ou linhas em uma determinada orientação, a medida que a rede se aprofunda as próximas camadas convolucionais são capazes de detectar combinações das formas encontradas pelas camadas anteriores como texturas e partes completas de objetos. Os filtros são parametrizados por **pesos** (*weights*) e o grande diferencial dessa técnica é que não é necessário que se defina quais pesos de cada filtro serão utilizados, eles são aprendidos pela rede automaticamente quando expostas a um grande volume de dados durante o treinamento, os pesos são iniciados com números aleatórios e a medida que o treinamento avança, os pesos são ajustados para melhorar o desempenho de cada camada (DESHPANDE, 2016).

A convolução é uma operação matemática bastante utilizada em processamento de imagens digitais com técnicas de filtragem no domínio espacial, por exemplo em filtros de suavização, aguçamento e detecção de bordas (GONZALEZ; WOODS, 2009). Consiste em aplicar um filtro sobre uma matriz, normalmente uma imagem, a fim de gerar uma nova matriz, similar a imagem original, porém com conteúdo diferente, mas produzido a partir dos valores originais.

O filtro é uma matriz 2D de dimensões reduzidas, comumente 2x2 ou 5x5, que possui a mesma profundidade da imagem original, os filtros contém pesos que são os

responsáveis pelo resultado que o filtro se destina. No início da operação, o filtro é comumente posicionado no canto superior esquerdo da imagem de modo a coincidir com os primeiros pixels da imagem, a área na imagem que o filtro sobrepõe a cada etapa recebe o nome de **campo receptivo**. Os pesos do filtro são então multiplicados pelos valores correspondentes na imagem, a soma dos resultados é inserida na primeira posição da matriz resultado, o filtro é então deslocado para a direita até o final da linha e depois é posicionado na próxima linha, e a operação se repete até o final da matriz, em um esquema de janela deslizante. O deslocamento do filtro a cada etapa é parametrizado pelo **passo** (*stride)*, quando o passo é 1 o filtro se deslocará 1 pixel por vez, dessa forma processará todos os pixels da matriz original, quando o passo é 2 o filtro se deslocará por 2 pixels, pulando 1 pixel, dessa forma processando metade de todos os pixels e assim por diante (DESHPANDE, 2016). Na Figura 5 pode-se ver uma ilustração que demonstra este processo.

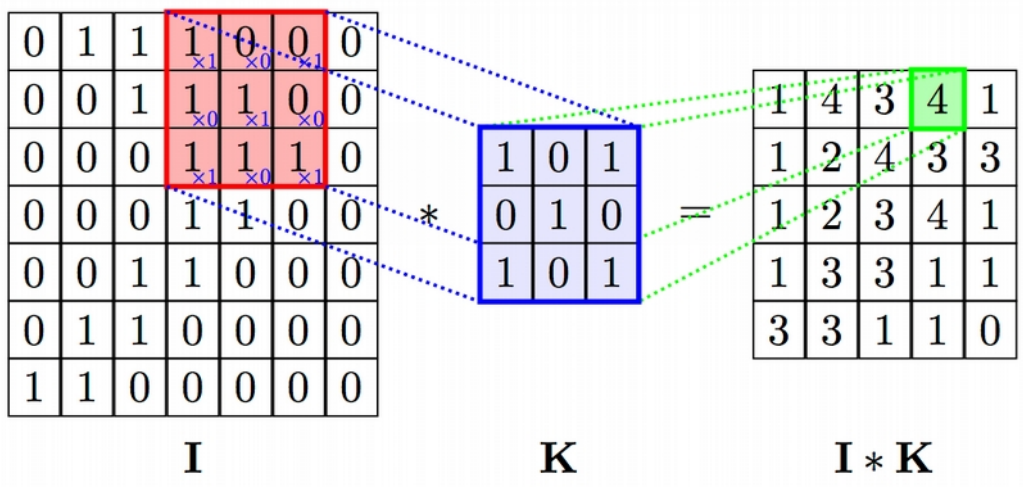



*Figura 6: A Convolução. (PETAR, 2019)*

Como resultado da aplicação do filtro, uma nova matriz é gerada com os valores resultado da convolução, que é chamada mapa de ativação, leva este nome porque contém as regiões que o filtro enfatizou, ou que foram ativadas pelos neurônios. As camadas convolucionais mais profundas possuem um campo receptivo proporcionalmente maior que as primeiras camadas.

#### 2.2.2.1.1.2 Camada de *Pooling*

A camada de *Pooling* tem por finalidade reduzir a dimensionalidade espacial do volume de matrizes (conjunto) da camada anterior, isto porque uma camada convolucional fornece como saída um volume de profundidade igual ao número de filtros desta camada. Por exemplo, uma imagem no sistema RGB (*Red, Green and Blue*) possui 3 canais de cor, dessa forma, uma imagem 30x30x3 pixels totalizam 2.700 pixels. Caso a camada convolucional possua 12 filtros, produzirá como resultado um volume de dimensões 30x30x12, totalizando 10.800 pixels, esse aumento expressivo nos dados também exige um maior poder de processamento. Dessa forma, a camada de *pooling* reduz a dimensionalidade espacial desse volume para compensar o aumento da profundidade, além disso, a redução de dimensionalidade ajuda as próximas camadas convolucionais a combinar atributos das camadas anteriores em estruturas mais elaboradas (DESHPANDE, 2016).

A técnica *maxpooling* é a mais comum entre as técnicas de *pooling*, consiste em aplicar um filtro, normalmente 2x2 com passo de mesmo tamanho sobre o volume de matrizes de entrada e retorna matrizes que contém o maior valor encontrado em cada sub-região da matriz de entrada durante o processo de convolução sobre todo o volume de matrizes (DESHPANDE, 2016). Na Figura 7 é possível ver como esse processo acontece.

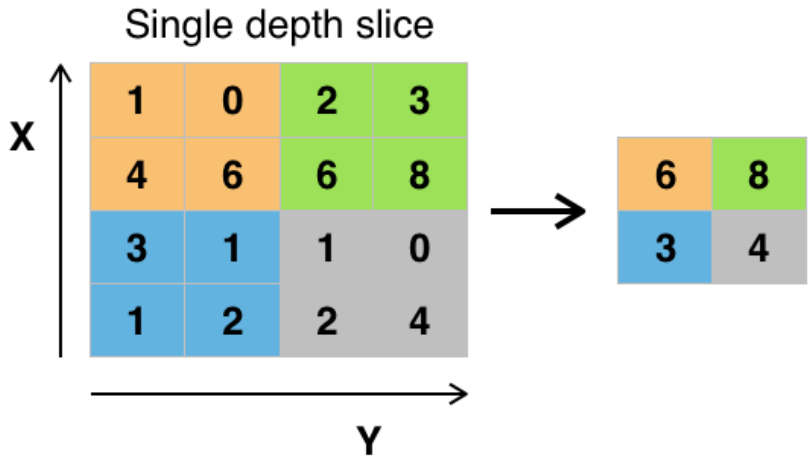



*Figura 7: Maxpooling. (DESHPANDE, 2016).*

#### 2.2.2.1.1.3 A Camada *Rectified Linear Units* (ReLU)

Após cada camada convolucional é comum aplicar uma camada não-linear, ou camada de ativação, existem muitos tipos de camadas de ativação, porém as ReLU são atualmente as mais usadas, isto porque observou-se que elas permitem que as redes treinem muito mais rápido sem comprometer a acurácia do modelo, já que exigem menos processamento que as demais.

A camada consiste em aplicar a função $f(x)=max(0,x)$ para todos os valores dos volumes de entrada, na prática ela substitui todos os valores negativos por 0 e mantém os valores positivos inalterados (DESHPANDE, 2016).

#### 2.2.2.1.1.4 Camada Completamente Conectada

O estágio final de uma ConvNet é composto pelas camadas completamente conectadas ou FC (*Fully Connected*). Como visto anteriormente, as camadas convolucionais são responsáveis por detectar atributos relevantes nas imagens e fornecer para as próximas camadas os mapas de ativação que contém a localização destes atributos relevantes. Já as FC, em tarefas de classificação, tem por objetivo determinar a que classe determinada imagem pertence. Elas fornecem como saída um vetor N-dimensional onde o N representa o número de possíveis classes. Este vetor contém a probabilidade de pertinência da imagem a todas as possíveis classes. Por exemplo, em um programa que pretende classificar dígitos de 0 a 9, o vetor terá 10 posições. Se o conteúdo do vetor for [.0, .1, .9, .1, 0, ... ], o primeiro valor (.0) refere-se ao dígito 0 e indica que a probabilidade da imagem representá-lo é 0%, na sequência, (.1) a probabilidade de ser um 1 é 10%, (.9) a probabilidade de ser um 2 é de 90%, como essa é a maior probabilidade essa é a resposta de classificação (DESHPANDE, 2016).

A primeira camada FC está totalmente conectada ao volume fornecido pela camada anterior, ela é composta por neurônios artificiais que estão completamente conectados entre si de maneira não linear. O que a camada FC faz é tentar correlacionar os

atributos apresentados no último mapa de ativação a uma determinada classe (DESHPANDE, 2016).

#### 2.2.2.2 *Transfer Learning*

Curry (2018) diz que "*Transfer learning* é um método de aprendizagem de máquina em que um modelo desenvolvido para uma tarefa é reusado como ponto de partida para outro módulo em uma tarefa diferente". O *transfer learning* difere de uma abordagem tradicional de aprendizagem de máquina porque utiliza um modelo pré-treinado, que fora desenvolvido para uma determinada tarefa, para encurtar o processo de treinamento de um novo modelo, destinado a uma tarefa distinta da anterior, usando este modelo como base para o início do treinamento do novo (PRAT; THRUN, 1997).

Os benefícios de utilizar essa técnica é que ela encurta o tempo e o esforço necessário para desenvolver e treinar um modelo partindo do zero, reutilizando peças ou módulos já desenvolvidos no modelo base, além disso requer menos amostras no *dataset* de treinamento que quando treina-se um modelo completamente novo (CURRY, 2018).

É um método bastante popular, Karpathy e Johnson (2019) dizem que:

> Na prática, poucas pessoas treinam uma ConvNet inteiramente do zero (com inicialização randômica), porque é relativamente raro ter um dataset de tamanho suficiente. Do contrário, é comum pré-treinar uma ConvNet em um dataset muito grande (ex.: ImageNet, que contém 1.2 milhões de imagens de 1000 categorias), e então usar a ConvNet treinada para inicializar outro modelo ou então utiliza-la como extrator de atributos na tarefa de interesse.

Isso é possível porque as primeiras camadas de uma ConvNet são treinadas para detectar formas elementares e simples, como tamanhos de objetos, linhas, bordas, texturas e tons. A medida que as camadas se aprofundam fazem combinações das estruturas detectadas nas camadas anteriores, o *transfer learning* reaproveita as primeiras camadas e otimiza as camadas finais para a tarefa de destino, economizando tempo e esforço no treinamento das primeiras camadas. É preciso

destacar ainda que para que a técnica seja possível, o modelo base precisa ser treinado com um *dataset* vasto e de propósito geral, menos específico que a nova tarefa, de preferência com características parecidas, para que se aproveite as camadas genéricas do primeiro modelo de forma eficiente (CURRY, 2018).

Existe diversas técnicas de *transfer learning,* a seguir serão apresentadas as mais comuns. *ConvNet Fixed Feature Extractor*: quando o modelo base é utilizado simplesmente como um extrator de atributos e a camada de classificação final é ignorada e substituída por um classificador linear como *Linear Support Vector Machines* (Linear SVM) ou *Softmax Classifier*. ConvNet Fine-Tuning: quando o modelo base é usado como partida e re-treinada usando um novo *dataset*, nessa abordagem todas as camadas podem ser atualizadas, porém é comum que só as camadas finais o sejam. *Pretrained Models*: quando *checkpoints* e pesos são disponibilizados para o uso de outras pessoas que podem decidir de que forma realizarão a técnica (KARPATHY; JOHNSON, 2019).

Uma ConvNet moderna pode exigir treinamentos longos de até 2-3 semanas, por isso é comum encontrar disponíveis na Internet modelos pré-treinados utilizando diversos tipos de arquitetura de rede sobre *datasets* gigantes e diversos, esses modelos podem ser utilizados livremente para o treinamento de novos modelos, os principais *frameworks* dedicados ao *deep learning* disponibilizam os *model zoo*, que são bibliotecas de modelos pré-treinados que podem ser utilizados livremente.

### 2.2.3 Classificação, Localização, Detecção e Segmentação

Em Aprendizagem de Máquina, os algoritmos podem ser agrupados quanto a finalidade do problema que se pretende resolver. No problema de classificação, ou reconhecimento, o objetivo é determinar a que categoria, ou tipo, um conjunto de dados pertence, o resultado da predição é um valor categórico (discreto e não ordenado), uma etiqueta que representa um grupo no conjunto de dados (HAM; KAMBER; PEI, 2011). Em trabalhos que envolvem imagens a finalidade é, com base no conteúdo da imagem, determinar a classe que ela pertence. Por exemplo, dada a

imagem de uma flor, determinar que espécie ela pertence.

No problema de localização, além de determinar a que classe pertence uma imagem, também é necessário determinar sua localização relativa na imagem utilizando uma caixa de fronteira (*bounding box*). Nos problemas de classificação e localização cada imagem contém apenas um objeto. Já no problema de detecção de objetos, todos os objetos na imagem precisam ser detectados, demarcados com caixas de fronteiras e etiquetados com a classe que o objeto pertence, então podem haver um ou muitos objetos de classes iguais ou diferentes (KARAGIANNAKOS, 2019) (PARMAR, 2018).

Para o problema de segmentação existem dois subtipos. A segmentação semântica tem por objetivo classificar todo e qualquer pixel da imagem para sua respectiva classe, diferente da detecção de objetos não se usa caixas de fronteiras, todos os pixels do objeto são marcados, não há distinção entre instâncias, dessa forma, em uma imagem que traz duas flores da mesma espécie, elas serão tratadas como o mesmo objeto. Já a segmentação de instância funciona como a segmentação semântica, porém faz distinção entre instâncias, por exemplo, uma imagem contém duas flores da mesma espécie, elas serão tratados como objetos do mesmo tipo, porém instâncias diferentes (KARAGIANNAKOS, 2019) (PARMAR, 2018).

Parmar (2018) diz ainda que "É bom destacar que estes termos não são claramente definidos na comunidade científica, então pode-se encontrar entendimentos diferentes sobre eles. Para mim, esta é a interpretação correta". Na figura 8 é possível ver as diferenças entre os problemas de visão computacional.

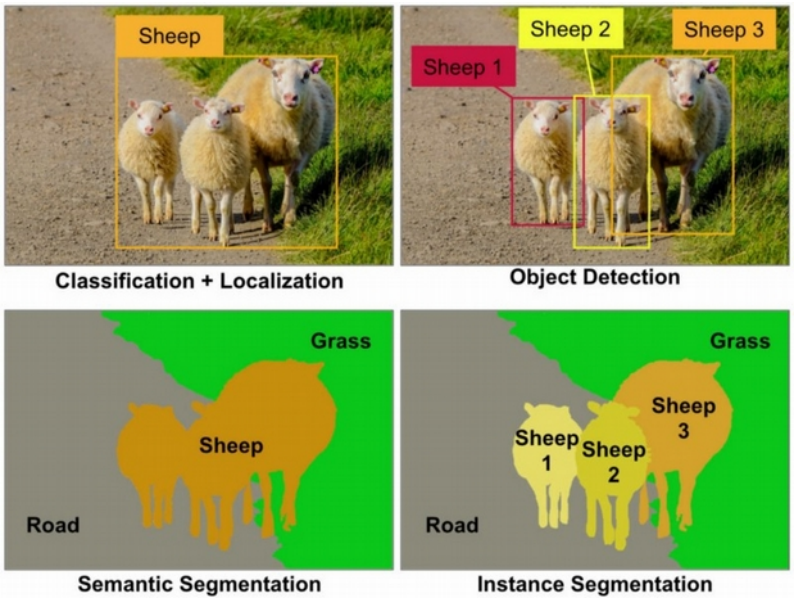



*Figura 8: Diferentes problemas de Visão Computacional (PARMAR, 2018).*

Neste trabalho, o problema é visto como uma tarefa de detecção de objetos, já que pretende-se a classificação e a localização de um ou mais glomérulos em imagens histológicas digitais, as ConvNets também são amplamente utilizadas para este fim.

### 2.2.4 Detecção de Objetos e as ConvNets

Como foi dito anteriormente, o problema de detecção de objetos é diferente do problema de classificação, para qual as ConvNets foram projetadas originalmente, principalmente porque na detecção de objetos é necessário localizar objetos que podem ter tamanhos diversos e quantidades variáveis para cada imagem, não sendo possível saber, de antemão, quais objetos e quantos estarão presentes em cada imagem. Por isso não seria possível utilizar uma ConvNet convencional, já que a última camada FC de uma ConvNet tem uma saída de tamanho pré-fixado (GANDHI, 2018).

Uma solução apressada para este problema poderia ser subdividir uma imagem em partes e aplicar uma ConvNet sobre cada uma delas em um esquema de janela deslizante, assim cada porção da imagem seria classificada individualmente, porém esta solução traz alguns problemas. Como não há como prever o tamanho dos objetos (escala) em cada imagem, seria necessária uma quantidade enorme de quadros com

várias opções de tamanho para conseguir detectar todos os objetos em diferentes escalas, além de utilizar um algoritmo para unir todos os resultados separados em detecções singulares, sem repetições, e isso é computacionalmente proibitivo já que exigiria muito tempo de processamento ou uma infraestrutura muito cara. Para contornar esses problemas, algoritmos dedicados à detecção de objetos estão sendo desenvolvidos constantemente, entre eles pode-se destacar: *Region Convolutional Neural Networks* (R-CNN) (GIRSHICK et al, 2014), *Fast* R-CNN (GIRSHICK, 2015), *Faster* R-CNN (REN et al, 2017) e You Only Look Once (YOLO) (REDMON et al., 2016).

Girshick et al. (2014) propôs o R-CNN, um método em que, inicialmente, para cada imagem seleciona 2.000 regiões, independentes de categoria, utilizando uma busca seletiva. Numa primeira etapa muitas regiões candidatas são selecionadas e em seguida agrupadas em regiões maiores considerando suas semelhanças, estas regiões são chamadas de regiões sugeridas (*region proposals*). Em seguida, o método utiliza uma rede ConvNet para extrair atributos das regiões sugeridas que são fornecidas como entrada para um classificador *Support Vector Machine* (SVM) que identificará os objetos nas regiões sugeridas selecionadas na etapa anterior. Este método foi um avanço para sua época, porém ainda é computacionalmente custoso, porque para o treinamento da rede é necessária a classificação das 2.000 regiões sugeridas para cada imagem, além disso, não pode ser utilizado para aplicações em tempo real porque necessita de 47 segundos para classificar uma imagem. Outro ponto é que a busca seletiva é um algoritmo fixo, ou seja, nada é aprendido nessa etapa, o que pode resultar em uma seleção de regiões pouco significativas (GANDHI, 2018). Na Figura 9 é possível ver uma imagem que ilustra o processo de busca seletiva.

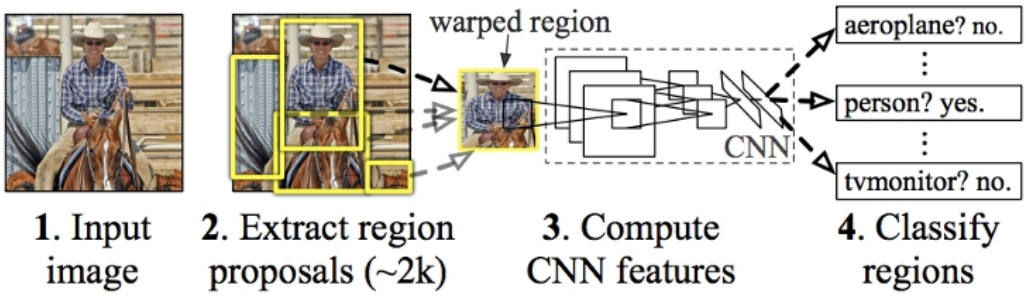



*Figura 9: R-CNN: Regiões com atributos CNN. (GIRSHICK et al. 2014)*

No ano seguinte os mesmos autores propuseram um melhoramento ao método anterior, o *Fast* R-CNN (GIRSHICK, 2015), aqui ao invés de fornecer as regiões sugeridas a uma rede ConvNet, a imagem inicial é inserida diretamente a uma rede ConvNet para que esta gere um mapa de atributos convolucionais. Em seguida, as regiões sugeridas são identificadas, colocadas em caixas de fronteira e redimensionadas para um tamanho fixo usando uma camada de *pooling Region of Interest* (RoI), para na sequência passarem por uma camada completamente conectada que classificará cada região. O motivo pelo qual este método é significantemente mais rápido que o anterior é que não é necessário executar convolução sobre as 2.000 regiões todas as vezes para cada imagem, a operação convolucional acontece somente uma vez, dela é gerada o mapa de atributos que é usado para as classificações das regiões (GANDHI, 2018).

Os métodos vistos anteriormente, o R-CNN e o *Fast* R-CNN, utilizam a busca seletiva para encontrar as regiões sugeridas, este algoritmo é lento e custoso computacionalmente, o que afeta a performance das redes. Para contornar esse problema Ren et al. (2017) propôs o *Faster* R-CNN, que elimina a necessidade de utilizar a busca seletiva e deixa que a rede aprenda as regiões sugeridas. De forma similar ao *Fast* R-CNN a imagem também é inserida em uma ConvNet para que seja gerado o mapa de atributos, porém a busca seletiva não é utilizada para identificar as regiões sugeridas, o mapa de atributos é fornecido para uma rede ConvNet separada e esta é encarregada de identificar as regiões sugeridas. Estas então são redimensionadas usando uma camada de *pooling* RoI que é então usada para as regiões sugeridas. Na Figura 10 é possível ver uma comparação do tempo de

processamento de uma imagem, em segundos, para cada método mencionado acima. Pode-se concluir que o método *Faster* R-CNN é muito mais rápido que os demais, tornando possível sua utilização em aplicações em tempo real (GANDHI, 2018).

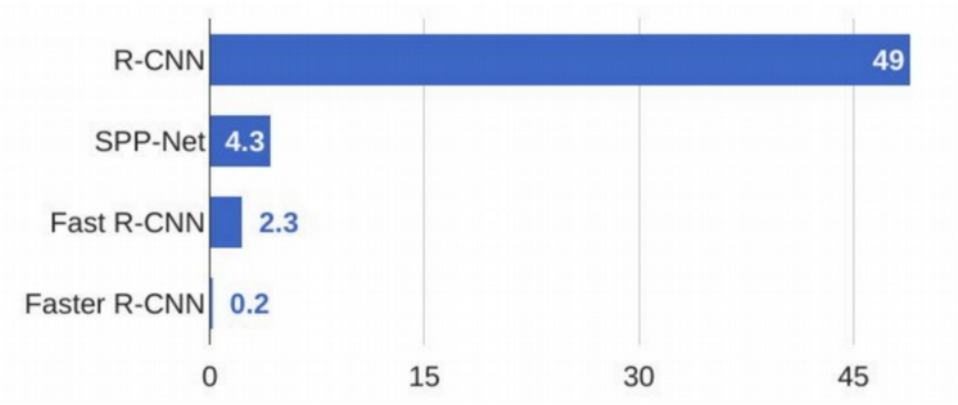



*Figura 10: R-CNN Teste de velocidade. (GANDHI, 2018)*

Todos os métodos apresentados até aqui utilizam regiões para localizar objetos na imagem, as redes não olham a imagem como um todo, utilizam regiões com alta probabilidade de conter objetos. O YOLO (REDMON et al., 2016) é um método de detecção de objetos bem diferente dos vistos anteriormente, nele uma única rede convolucional identifica as caixas de fronteira e as probabilidades de classificação para cada caixa.

No YOLO a imagem é dividida em uma grade $SxS$ e uma quantidade $m$ de caixas de fronteiras são identificadas para cada unidade da grade. Para cada caixa de fronteira, a rede retorna a probabilidade de pertinência às classes e valores de *offset* para a caixa. As caixas de fronteiras que tiverem uma probabilidade acima de um limiar pré-estabelecido são selecionadas e utilizadas para detectar os objetos na imagem. Na Figura 11 é possível ver uma ilustração deste processo.

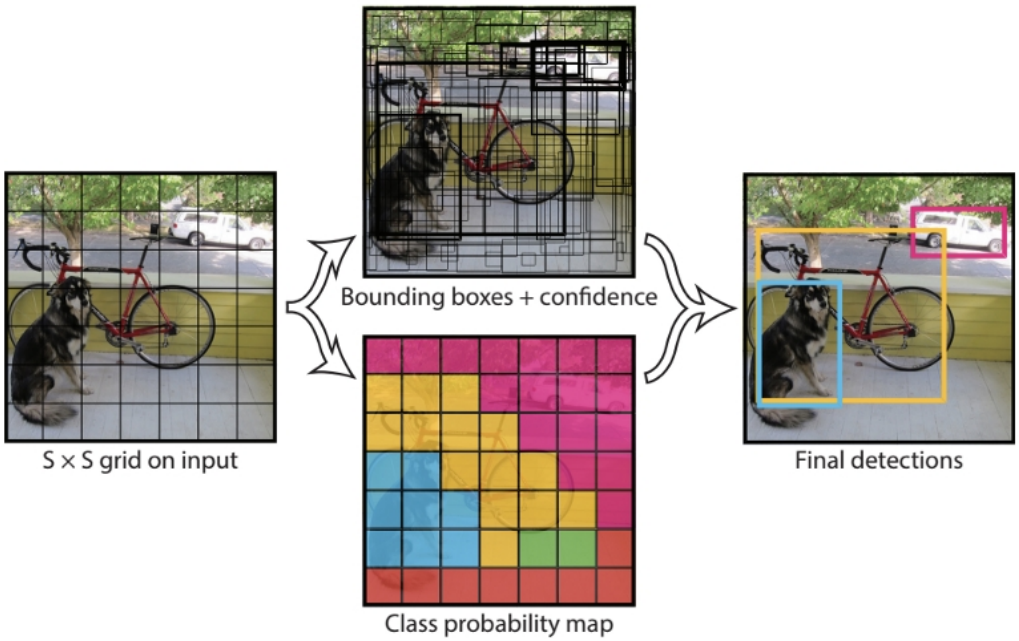



*Figura 11: YOLO. (REDMON et al., 2016)*

O YOLO é muito mais rápido (45 *frames* por segundo) do que os outros métodos de detecção de objetos, porém tem limitações na detecção de objetos muito pequenos, por exemplo, teria dificuldades em detectar bandos de pássaros (GANDHI, 2018). Além disso, ainda não houve tempo suficiente de popularizar suas implementações, até o momento só existe uma implementação que é incompatível aos populares *frameworks* dedicados ao *deep learning*.

### 2.2.5 Arquiteturas de Redes Convolucionais

As arquiteturas de redes convolucionais determinam a estrutura geral de uma rede, ou seja, a quantidade de camadas e como devem estar conectadas entre si. A maioria das ConvNets combinam suas camadas num arranjo em cadeia, onde cada camada está em função de uma camada anterior. Nestas redes, as características mais importantes são a profundidade (*deep*), ou seja, quantas camadas possui a redee, e a largura de cada camada (*width*), ou seja, quantos neurônios ou filtros deve possuir cada camada (GOODFELLOW; BENGIO; COURVILLE, 2016).

Até mesmo uma rede com uma única camada pode ser suficiente para se ajustar ao conjunto de treinamento. Redes mais profundas conseguem utilizar menos neurônios além de conseguir generalizar melhor resultados ao conjunto de testes, porém são

mais difíceis de ajustar. A rede ideal para determinada tarefa deve ser encontrada por meio de experimentação guiada pelo monitoramento do erro no conjunto de validação (GOODFELLOW; BENGIO; COURVILLE, 2016).

A seguir serão apresentadas as arquiteturas das ConvNets utilizadas neste trabalho: *Inception*, ResNet (*Residual Networks*) e SSD (*Single Shot Multibox Detector*).

#### 2.2.5.1 Rede *Inception*

A rede *Inception* (SZEGEDY, 2014) foi posta a prova na competição e *ImageNet Large-Scale Visual Recognition Challenge* de 2014 (ILSVRC 2014) (RUSSAKOVSKY et al., 2015) e redefiniu o estado da arte do reconhecimento e detecção de objetos. Sua arquitetura conta com 27 camadas e 5 milhões de parâmetros, que são números modestos em comparação a seus concorrentes, é uma rede escalável, ou seja, pode ser aumentada conforme a necessidade e a disponibilidade de infraestrutura. Seu principal diferencial é que ela tem suas camadas organizadas de forma que são pouco conectadas.

Comumente, quando se tenta aumentar a performance de uma *ConvNet* duas estrategias são empregadas: aumentar a profundidade (*deep*), ou seja, a quantidade de camadas da rede, ou aumentar a largura da rede, o que significa aumentar a quantidade de filtros ou neurônios de cada camada, porém essas estratégias trazem implicações negativas. O aumento da profundidade da rede tende a aumentar o sobre-ajustamento (*overfitting*), que acontece quando um modelo decora as características de um *dataset* em particular. Quando isso acontece a performance do modelo cai quando ele é exposto a amostras inéditas, diminuindo sua capacidade de generalizar, perdendo em performance. Isso acontece especialmente com *datasets* com poucas amostras, para muitos casos, aumentar o número de amostras do *dataset* é muito caro ou impossível. A outra estratégia, a de aumentar a largura da rede, fará aumentar o número de parâmetros, o que exigirá mais poder de processamento e recursos computacionais (SZEGEDY et al., 2014).

Para contornar tais problemas os autores sugeriram uma arquitetura de rede pouco conectada (*sparcely connected*) e menos profunda, em contrapartida é mais larga e modular. As camadas principais são compostas por módulos *Inception*, que são compostos por camadas convolucionais e de *pooling* que processam a entrada paralelamente e concatenam os resultados em uma única saída que servirá de entrada a próxima camada da rede ou módulo *Inception*. Na Figura 12 é possível ver a arquitetura da rede *Inception* e na Figura 13 vê-se a arquitetura de um módulo *Inception* (SZEGEDY et al., 2014) (SHAIK, 2018).

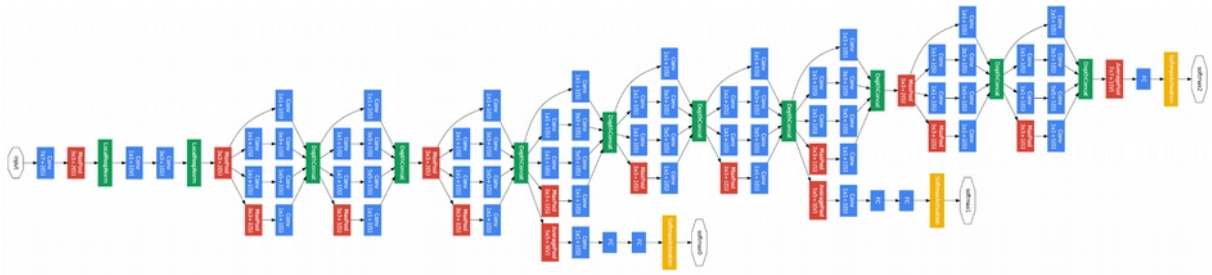
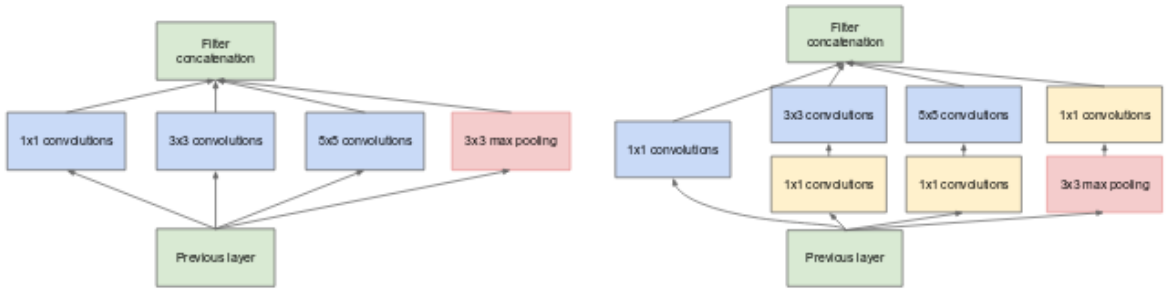


*Figura 12:* Rede *Inception (SHAIK, 2018)*




(a) Inception module, naïve version (b) Inception module with dimension reductions

*Figura 13: Módulo Inception (SZEGEDY et al, 2014)*

No ano seguinte, os autores propuseram melhoramentos à rede *Inception* com duas novas versões chamadas *Inception* V2 e V3, no mesmo artigo (SZEGEDY et al., 2015) que traziam melhor acurácia e a redução da complexidade computacional.

A premissa da rede *Inception* V2 é evitar a redução drástica da dimensionalidade espacial das amostras, o que causa perda de informação e compromete a acurácia da rede, A solução foi trocar as camadas convolucionais 5x5 por duas camadas 3x3, o que contribuitivamente torna a rede também mais rápida. A versão 3 da rede

funciona da mesma forma que a V2 só que emprega técnicas para tentar melhorar seus resultados sem mudar drasticamente a arquitetura anterior, emprega otimizadores *RMSProp (Root Mean Square Propagation)*, camadas convolucionais 7x7 e classificadores auxiliares *BatchNorm*. A rede *Inception* V2 foi utilizada neste trabalho (SHAIK, 2018) (RAJ, 2018).

#### 2.2.5.2 Redes Residuais

Como já visto, uma das estratégias para aumentar a performance de uma rede é aumentar a profundidade dela, porém esta estratégia tem limitações. Quando uma rede tende a aumentar sua profundidade, além de ter de contornar o *overfitting* com o aumento da quantidade de amostras de treinamento, a partir de certo ponto sua acurácia começa a degradar-se e cai rapidamente, dessa forma não é possível aumentar a profundidade de uma rede indefinidamente.

Diante destes problemas, He et al. (2015) propuseram as *Residual Networks* (ResNet), uma metodologia tão eficiente que venceu o concurso ILSVRC de 2015 (RUSSAKOVSKY et al., 2015) em classificação e detecção de objetos e venceu também o *Common Objects in Context* de 2015 (COCO 2015) (LIN et al., 2014) em detecção e segmentação de objetos (JAY, 2018) (PEIXEIRO, 2019).

Redes neurais artificiais muito profundas são difíceis de treinar e são mais suscetíveis a *vanishing gradients*, que ocorre quando a derivada da função de erro e tão pequena que a atualização dos pesos não acontece, ou *exploding gradients*, que é o oposto, quando as derivadas são muito grandes a ponto de extrapolar a capacidade de representação numérica dos pesos. Para resolver esse problema, a unidade de ativação de uma camada pode alimentar diretamente uma camada mais profunda da rede, pulando uma camada, o que é chamado de salto de conexão (*skip connection*), esse recurso é a base das redes residuais ou ResNets (PEIXEIRO, 2019).

As ResNet são compostas por blocos residuais ou blocos identidade, que simplesmente passam os valores recebidos para a próxima camada sem alterar seus valores. Na

Figura 14 é possível ver sua anatomia.

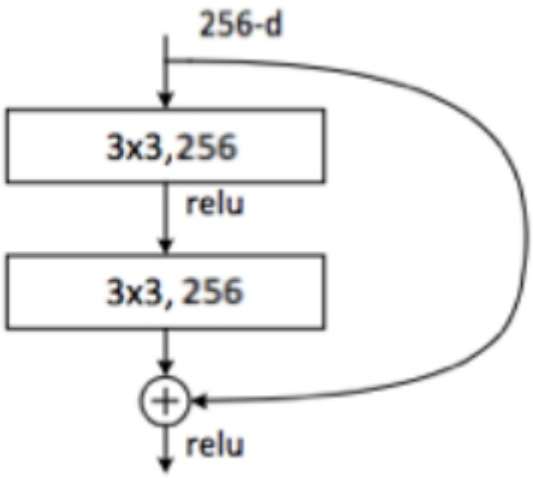



*Figura 14: Bloco Residual (PEIXEIRO, 2019)*

As ResNet não sofrem com a limitação de profundidade das redes convencionais e graças aos blocos residuais elas são capazes de treinar com um número de camadas muito maior que os outros tipos de rede, a medida que a profundidade aumenta o erro tende a cair. As ResNet tornaram possível treinar redes com mais de 100 camadas, podendo alcançar 1000 camadas ou mais (PEIXEIRO, 2019).

#### 2.2.5.3 Rede Single Shot Multibox Detector (SSD)

As rede SSD (LIU et al., 2016) tem por principal característica a utilização de uma única rede ConvNet que combina a geração de regiões sugeridas e extração de características. A rede aplica um conjunto padrão de caixas de tamanhos e aspectos diferentes sobre todos os mapas de atributos. Enquanto faz a predição, a rede gera *scores* (confiança) para a presença de cada classe de objeto em cada caixa padrão e faz ajustes para que a caixa se encaixe melhor ao formato do objeto. A rede elimina a necessidade de gerar regiões sugeridas, já que usa os conjuntos de caixas padrão, combinando diferentes etapas em uma única rede ConvNet, o que a torna bastante rápida na detecção de objetos, capaz de realizar 59 *frames per second* (FPS) com imagens 300x300 pixels (HULSTAERT, 2018).

### 2.2.6 Análise de Desempenho

No problema de detecção de objetos é preciso localizar todos os objetos presentes na imagem, cada objeto deve ser delimitado por uma caixa de fronteira e cada caixa deve possuir uma etiqueta indicando a que classe pertence (KARAGIANNAKOS, 2019) (PARMAR, 2018). Por conta destas características, as métricas utilizadas para este problema são diferentes das utilizadas, por exemplo, para o problema de classificação. A métrica de avaliação padronizada pelas competições de detecção de objetos (RUSSAKOVSKY et al., 2015) (LIN et al., 2014) (GIRSHICK et al., 2014) (REN et al., 2017) é o "*mean average precision*" (mAP) que é um número de 0 a 100 onde os maiores valores são os melhores. Cada caixa de fronteira traz um *score*, também chamado de confiança (*confidence*), que indica o nível de certeza na classificação do objeto contido nela. Baseado nestes valores uma curva *precision-recall* (*PR curve*) é calculada para cada classe de objeto variando um limiar de confiança. A "*average precision*" (AP) é a área que fica abaixo dessa curva. Primeiro a AP é calculada para cada classe e então é calculada a média de todas as classes. O resultado final é o mAP (HULSTAERT, 2018).

Cada resultado de detecção só é aceito como um verdadeiro positivo se houver a sobreposição entre a área da caixa encontrada com a caixa de fronteira de referência (*ground-truth*), para aferir essa situação é calculado o "*intersection over union*" (IoU) que é usualmente aceito acima do 0.5. Algumas competições exigem medidas de mAP por faixa de IoU, como por exemplo, mAP@0.25 ou mAP@0.5 que respectivamente refere-se ao IoU mínimo de 0.25 e 0.5 (HULSTAERT, 2018). Na Figura 15 vê-se como é calculado o IoU, sobre a fotografia a caixa em vermelho representa o resultado da predição, a caixa em azul é a caixa referência, a área da sobreposição entre as duas caixas é dividida pela área da união das mesmas caixas resultando no IoU.

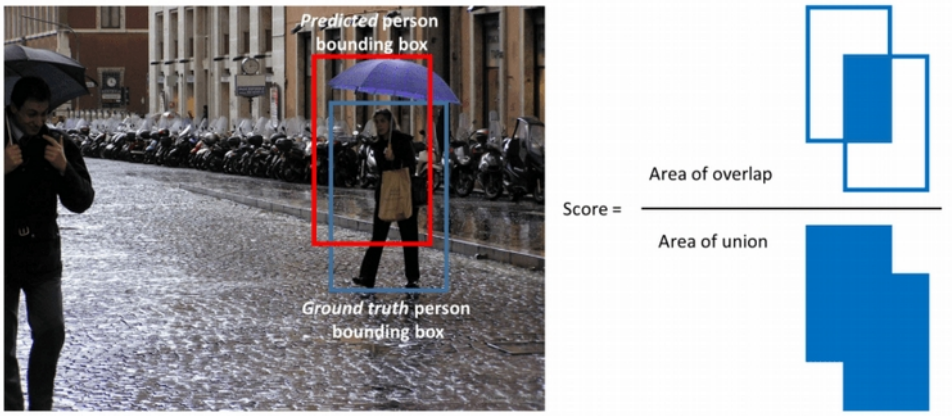



*Figura 15: Cálculo do IoU (HULSTAERT, 2018)*

Apesar do mAP ser o padrão das competições de detecção de objetos, alguns trabalhos apresentam seus resultados utilizando as métricas comuns em trabalhos de classificação, são elas: *Precision*, *Recall* e *F1 Score*.

*Precision* (Valor Preditivo Positivo) – É a proporção de verdadeiros positivos (VP) sobre todas as predições positivas, ou seja, verdadeiro positivos e falsos positivos (VP + FP), em resumo, das predições positivas feitas quantas acertou. No contexto deste trabalho, *precision* é afetada por casos de detecção de estruturas do tecido que não são glomérulos, mas foram detectadas como se fossem. Dado por:

$$Precision = VP / (VP + FP)$$

*Recall* (Sensibilidade) – É a proporção de verdadeiros positivos (VP) sobre a soma dos verdadeiros positivos com os falsos negativos (FN), ou seja, é a capacidade que o classificador tem de acertar a classe positiva. No contexto deste trabalho, o *recall* é afetado por casos em que glomérulos deixaram de ser detectados pelo modelo. Dado por:

$$Recall = VP / (VP + FN)$$

*F1 Score* – É a média harmônica entre *Precision* (P) e *Recall* (R), utilizada para comparar o desempenho geral de classificadores. Dado por:

$$F1 = 2 * (P * R / P + R)$$

# **Capítulo 3** Trabalhos Relacionados

Este capítulo tem o objetivo de apresentar os artigos disponíveis na literatura que tratam do tema relacionado a este trabalho.

Gallego et al. (2018) propuseram um método de classificação e detecção de glomérulos em imagens WSI (*whole-slide image*, ou lâminas histológicas integralmente digitalizadas), as quais são imagens de altíssima resolução (gigapixels), que podem conter dezenas de glomérulos reduzindo falsos positivos e falsos negativos, utilizando técnicas de *deep learn*. Eles treinaram redes *CNN* para que fossem capazes de identificar duas classes de objetos, glomérulos e não-glomérulos a fim de identificar segmentos que continham regiões contendo glomérulos. Utilizaram *transfer learn* a partir de uma rede AlexNet pre-treinada para adaptá-la ao problema proposto. O método utiliza um algoritmo de janela deslizante onde a cada passo a rede CNN é empregada para classificar se existem ou não glomérulos no segmento selecionado. Utilizaram também pré-processamento para normalização de cor, na tentativa de reduzir a variação de cores em imagens inter laboratórios. O método conseguiu 0.881 de *precision*, 1.0 de *recall* e 0.937 de *F1 Score*. Os autores dizem que os resultados indicam que o método é eficiente na detecção de glomérulos em imagens WSI.

Simon et al. (2018) propuseram um método de detecção automática de glomérulos em imagens WSI. O método proposto utiliza técnicas de aprendizagem de máquina, extraindo o vetor de atributos das imagens utilizando mrcLBP (*Multi Radial Color LBP)*, uma adaptação do descritor visual *LBP* (*Local Binary Pattern). E*sses atributos são usados para treinar um classificador SVM, e após o treinamento, o modelo é então aplicado sobre as imagens WSI na tentativa de determinar a localização dos glomérulos. Os autores utilizaram amostras de ratos, camundongos e

humanos, utilizaram 7 *datasets* distintos para o treinamento, contendo imagens com glomérulos e imagens sem glomérulos, colorados com H&E, Jones, PAS e Gomorri Trichrome, utilizaram tanto imagens de glomérulos saudáveis como glomérulos doentes, totalizando 7.099 imagens contendo glomérulos e 15.750 imagens sem glomérulos. Utilizaram 5 CPUs Intel Core I7 com 40 Gb de memória RAM (*Random Access Memory*). O método proposto consegue mais de 0.9 de precisão, e mais de 0.7 de r*ecall*.

Sarder, Ginley e Tomaswewski (2017) propuseram um método de detecção automática das bordas dos glomérulos. O método usa uma abordagem integrada utilizando filtros de Gabor, Gaussian Blurring, estatística F-Testing e transformação de distâncias. O método proposto é um aprimoramento do método de segmentação por textura Gabor. Consegue acurácia e precisão médias de 0.89 e 0.97, respectivamente quando utilizadas 200 imagens de glomérulos coradas com Hematoxilina e Eosina (H&E) e acurácia e precisão médias de 0.88 e 0.94, respectivamente quando utilizando 200 imagens de glomérulos coradas com PAS. Foram utilizadas imagens histológicas extraídas de ratos saudáveis. Os autores concluíram que o método proposto pode trazer avanços na análise estrutural clínica de glomérulos e ajudar a conceber diagnóstico e intervenções em tempo real.

Ginley et al. (2017) propuseram um método de classificação automática não-supervisionada de glomérulos em imagens histológicas utilizando filtros de Gabor e testes estatísticos. Argumentam que filtros de Gabor são muito utilizados para analisar imagens histológicas de outros órgãos. Os autores informam já ter trabalhado anteriormente com filtros de Gabor para detecção dos glomérulos, sendo este trabalho um aprimoramento do método anterior. Inicialmente, o método utiliza uma análise de *hotspot* para selecionar áreas candidatas para uma análise mais detalhada posteriormente, essa estratégia é vastamente utilizada em trabalhos similares. Em seguida, os contornos dos glomérulos são refinados através de uma combinação de métodos como: segmentação de textura de Gabor, *Gaussian Blurring, F-testing* para o espaçamento intra-glomerular, mapeamento espacial ponderado para enfatizar a

concentração glomerular. O método é capaz de identificar os glomérulos utilizando os cinco corantes mais comuns: H&E, PAS, Gömöri's trichrome, Congo red (CR), e Jones Silver. Os autores conseguiram resultados com médias de sensibilidade/especificidade 0.88/0.96 e acurácia de 0.92 em 1000 imagens renais de ratos. Eles concluíram dizendo que o método proposto abre as portas para a análise automática não-supervisionada estrutural de glomérulos ao remover o obstáculo da identificação automática de glomérulos. O principal avanço do trabalho foi descobrir que a discriminação de textura é altamente eficiente na identificação das bordas dos glomérulos independente de qual corante está sendo utilizado.

Sarder, Ginley e Tomaskewski (2016) propuseram um método automático de identificação de características patológicas relevantes em lâminas histológicas renais de ratos saudáveis. O método parte de imagens de lâminas recortadas contendo de 10-15 glomérulos, coradas com PAS. As imagens então são convertidas em escala de cinza para evidenciar regiões de alta densidade de núcleos, um mapa de calor é criado a partir da imagem para destacar as áreas com maior densidade. As áreas identificadas são então envoltas em caixas de fronteira, as áreas delimitadas por elas são expandidas com 250 pixels para todas as direções. Essas regiões são cortadas da imagem original e salvas em novas imagens, contendo glomérulos individuais. Foram usadas 15 imagens de biópsias de ratos sadios completas, que continham 148 glomérulos. Conseguiram como resultado uma acurácia média de 0.88 na detecção de glomérulos. Concluíram que o método traz grande potencial de incrementar informações disponíveis durante os diagnósticos clínicos.

Marée et al. (2016) propuseram uma metodologia para detecção de glomérulos que não exige a padronização do protocolo de captação das imagens histológicas digitalizadas. Os autores afirmam que a metodologia independe do corante utilizado, argumentam que as metodologias utilizadas até então exigem que o banco de imagens fosse captado por um único laboratório, seguindo um protocolo de aquisição padronizado, a fim de reduzir a variação de cor, já que esse fator atrapalha os algoritmos automáticos de detecção. Utilizaram dois *softwares* de código aberto, o

Cytomine que é destinado a análise colaborativa e compartilhamento de imagens de alta resolução e o Icy, uma plataforma colaborativa que utiliza a linguagem Java e é destinado a analise de bio-imagens, este também é uma comunidade de compartilhamento de algoritmos e protocolos dessa área. O processo utiliza os algoritmos WND-CHARM (*Multi-purpose image classification using compound transforms*) e ET-FL (*Extremely Randomized Trees for Feature Learning*) para a identificação de áreas de interesse, o que segundo eles não demonstrou ser eficiente. Para contornar esse problema, propuseram técnicas de normalização de cor a fim de melhorar os resultados anteriores. O banco de imagens utilizado possuía 200 imagens de lâmina completa de tecido humano, cada imagem continha além de glomérulos outros tipos de tecidos e estruturas. Utilizaram 100 imagens de lâmina para o treinamento, contendo um total de 2.927 glomérulos e 13.648 não glomérulos. As imagens restantes foram utilizadas para a avaliar a performance do método. Os autores conseguiram o melhor resultado utilizando o algoritmo ET-FL, combinado com a normalização de cor, alcançando 0.76 de acerto ao identificar glomérulos e 0.99 de acerto ao identificar os não-glomérulos. Eles concluíram que o método ainda não é satisfatório para uso prático na patologia renal.

Kato et al. (2015) propuseram um método que define um novo descritor para detecção de glomérulos em imagens histológicas. Utilizaram imagens de lâminas completas digitalizadas de seções renais de ratos, imagens de grandes dimensões, cada imagem podendo ter o tamanho de $10^8$ pixels e centenas de glomérulos ao mesmo tempo. Afirmam que o modelo deriva da técnica R-HOG (*Retangular Histogram of Oriented Gradients*), que quando aplicada resulta em muitos falso positivos por ser pouco flexível na definição das bordas. A nova técnica proposta, S-HOG (Segmental *Histogram of Oriented Gradients*), é mais flexível e reduziu a quantidade de falsos positivos pela metade. O método consiste em três etapas, o pré-rastreamento, segmentação e classificação. Em cada etapa é utilizado o algoritmo SVM com diferentes tipos de descritores HOG (*Histogram of Oriented Gradients*), resultando em três SVMs no total. O método conseguiu uma média de 0.866, 0.874 e 0.897 para

*f-measure*, precisão e *recall* respectivamente. Foram utilizadas 20 imagens de seção completa, com tamanho médio de 9.849 x 20.944 pixels. Também desenvolveram e propuseram um novo algoritmo de segmentação, o DCDP (do inglês *Divide & Conquer Dynamic Program*). Os autores concluíram que o novo descritor incrementa a eficiência do método anterior (R-HOG) e que o algoritmo criado por eles, o DCDP, é mais rápido que o algoritmo de segmentação anterior (EDP – *Exhaustive Dynamic Program*).

Zhang, Hu e Zhu (2011) propuseram um método automático de extração de contorno dos glomérulos utilizando algoritmos genéticos para preencher as bordas faltantes que são removidas na etapa inicial de remoção de ruído das imagens, processo que descontinua o contorno original dos glomérulos. Utilizaram o operador de Canny para detectar as bordas, *labeling* para destacar o contorno, *thinning* e *cross point deletion* para remover os ruídos, como resultado alguns contornos podem ser removidos pela etapa de remoção de ruído, o algoritmo genético é utilizado para recriar as bordas originais removidas. Foram utilizadas 100 imagens de glomérulos, não especificando de qual espécie, processados em computadores de CPU com 1,83 GHz e MATLAB 7.0. Os autores concluíram que o método proposto conseguiu alta eficiência sem sacrificar a acurácia, porém não apresentaram métricas objetivas dos resultados alcançados.

# **Capítulo 4** Materiais e Métodos

Neste capítulo serão apresentados os materiais e métodos que foram utilizados para a implementação deste trabalho. Inicialmente, serão apresentados os recursos, materiais e infraestrutura necessários para a implementação do trabalho, a seguir, como se deu a criação dos *datasets*, logo após, a configuração do roteiro e o treinamento dos modelos, por fim como se deram os testes de validação e testes finais do modelo. A Figura 16 traz um esquema que representa as etapas dessa metodologia.

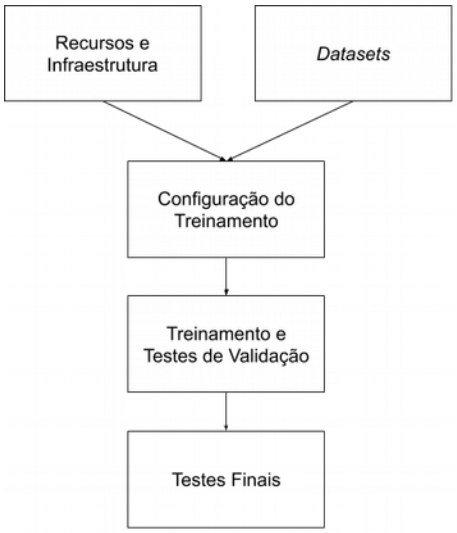



*Figura 16: Metodologia*

### 4.1.1 Recursos e Infraestrutura

A implementação do trabalho foi realizada utilizando a linguagem de programação *Python* (ROSSUM, 1995), versão 2.7.14, foi utilizado um *framework* dedicado ao *deep learning*, o *Tensorflow* (ABADI et al. 2015), versão 1.12.0. Também foi utilizado um *framework* dedicado a detecção de objetos, o TOD (*Tensorflow Object Detection API*) (HUANG et al. 2017), versão 2c181308 052361de. Durante a confecção dos *datasets* as imagens tiveram de ser etiquetadas com caixas de fronteira indicando a localização e limites dos glomérulos em cada imagem, essa tarefa foi feita utilizando a

ferramenta *LabelImg* (TZUTALIN, 2015).

Em razão da necessidade de alto desempenho computacional e das limitações financeiras deste trabalho, uma arquitetura de *hardware* na nuvem foi utilizada, o *Google Compute Engine*. Uma máquina virtual foi criada nesta plataforma e operada remotamente durante todo o tempo. A maquina escolhida possui 8 vCPU ( *Virtual Computer Process Unit*), as vCPUs são núcleos de processadores físicos, porém alocados dinamicamente, conforme a necessidade e a disponibilidade no *datacenter* onde as máquinas ficam alocadas fisicamente, todos os processadores são modelo *Intel Xeon*, mas de especificações diversas podendo variar o *clock* de 2,2 giga-hertz a 3,6 giga-hertz. O *datacenter* escolhido está localizado em Los Angeles, Califórnia, Estados Unidos. Este *datacenter* foi escolhido em razão da disponibilidade de placa aceleradora de vídeo. A máquina conta com 30 *gigabytes* de memória RAM (*Random Access Memory*), 50 *gigabytes* de disco de armazenamento SSD (*Solid State Drive*) e uma placa aceleradora de vídeo *Nvidia Tesla*. O sistema operacional utilizado foi o *Ubuntu Linux* versão 16.04, e os demais *softwares* já foram mencionados anteriormente.

## *4.2 Datasets*

Os *datasets* utilizados nesse trabalho foram criados a partir de um conjunto de imagens fornecidas pelo médico patologista Dr. Washington Luis Conrado dos Santos, do Centro de Pesquisas Gonçalo Muniz da Fundação Oswaldo Cruz (CpqGM/FIOCRUZ). As imagens foram obtidas através de fotografia digital, utilizando uma câmera fotográfica digital acoplada ao microscópio. Foram utilizadas lâminas histológicas de tecidos humanos saudáveis e de tecidos acometidos por glomerulosclerose segmentar e glomerulopatia membranosa, imagens de tecidos corados com H&E e PAS.

O conjunto original é composto por 466 imagens de glomérulos saudáveis, 720 imagens de tecidos com glomérulos afetados por glomeruloscleroses segmentar e 869 imagens de tecidos afetados por glomerulopatias membranosas, totalizando 2055

imagens. As imagens foram armazenadas em formatos diferentes: JPEG (*Joint Photographic Experts Group*), GIF (*Graphics Interchange Format*) e TIFF (*Tagged Image File Format*), todas no modelo de cor RGB. A vasta maioria está no formato JPEG e poucas imagens nos formato GIF e TIFF. As imagens foram capturadas em diversas escalas de aproximação, em algumas pode-se ver o glomérulo ocupando boa parte da imagem, trazendo todos seus detalhes de textura, enquanto em outras imagens eles aparecem ocupando áreas menores, em grupos de alguns glomérulos ou um único glomérulo afastado, trazendo poucos detalhes da textura de cada glomérulo. A resolução espacial média das imagens é de 0.798 Megapixel, a resolução máxima é de 3.145 Megapixel e a mínima é de 0.2983 Megapixel. A imagem com maior resolução possuía 2048x1536 *pixels* e a com menor resolução 632x472 *pixels*.

A partir do conjunto original de imagens, com o auxilio de um médico patologista, foram selecionadas imagens para a criação de três *datasets:* treinamento, validação e testes. Nenhum destes *datasets* compartilhou imagens entre si.

Após a divisão dos sub conjuntos, todas as imagens foram anotadas, utilizando caixas de fronteira (*boundary boxes*) para delimitar a localização e os limites de cada glomérulo nas imagens. Este processo foi feito manualmente utilizando o software *LabelImg*. Após a anotação, as caixas foram salvas em arquivo XML (*Extensible Markup Language*) no formato PASCAL VOC (*Pascal Visual Object Classes*) (EVERINGHAM et al., 2015), utilizado na competição *ImageNet (RUSSAKOVSKY et al., 2015)*, um arquivo para cada imagem. Na Figura 17 é possível ver um exemplo como este processo foi feito.

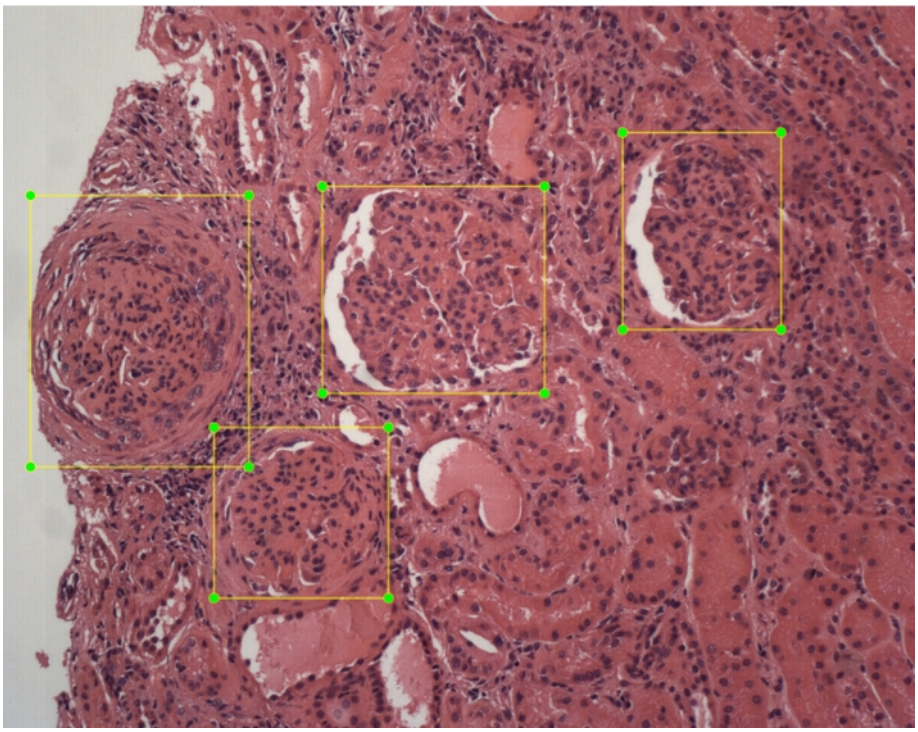


*Figura 17: Imagem anotada usando o software LabelImg.*

### 4.2.1 *Dataset* de Treinamento

Para este *dataset* foram utilizadas 509 imagens anotadas, sendo 209 imagens contendo glomérulos saudáveis, 150 imagens contendo glomérulos afetados por glomerulopatias membranosas e 150 imagens contendo glomérulos afetados por glomeruloscleroses segmentar.

### 4.2.2 *Dataset* de Validação

Diferentemente do *dataset* de treinamento, o *dataset* de validação tem por finalidade testar o desempenho dos modelos a medida que seu treinamento avança. Foram utilizadas 200 imagens, sendo 100 imagens contendo glomérulos saudáveis, 50 imagens contendo glomérulos afetados por glomerulopatias membranosas e 50 imagens contendo glomérulos afetados por glomeruloscleroses segmentar.

### 4.2.3 *Dataset* de Testes

O *dataset* de testes não é utilizado no processo de treinamento do modelo, seu objetivo é testar o modelo após o treinamento, apresentando amostras nunca vistas

antes por este, a fim de avaliar sua capacidade de generalização. Foram utilizadas 200 imagens, sendo 100 imagens contendo glomérulos saudáveis, 50 imagens contendo glomérulos afetados por glomerulopatias membranosas e 50 imagens contendo glomérulos afetados por glomeruloscleroses segmentar.

### 4.2.4 Conversão dos *Datasets*

Após a construção dos *datasets* eles precisaram ser convertidos para o formato aceito pelo *TensorFlow,* chamado de *TFRecord.* Neste formato todas as imagens juntamente com as etiquetas das caixas de fronteiras e classes são concatenadas em um único arquivo, de acordo com os criadores da tecnologia, isso possibilita otimizar o desempenho da ferramenta, paralelizando o processo de treinamento e tornando possível a utilização de placas aceleradores de vídeo, *Graphics Processing Unit* (GPU). Dessa forma, os *datasets* de treinamento, validação e testes foram convertidos para esse formato gerando 3 arquivos, o arquivo de treinamento ocupando 96 *megabytes*, o de validação 52 *megabytes* e o da testes 55 *megabytes.*

## 4.3 Configuração do Treinamento

O TOD recomenda a utilização do *Transfer Learning* (PRAT; THRUN, 1997) para o treinamento dos modelos, esta técnica já apresentada na seção 2.2.6, consiste em utilizar um modelo previamente treinado como base para o treinamento de um novo modelo, aproveitando as camadas mais rasas da rede e ajustando as camadas mais profundas para a nova tarefa. Para isso, ajustes são feitos, e um novo *dataset* é apresentado ao modelo, fazendo com que este adapte-se ao novo problema. As vantagens dessa abordagem são a considerável economia de tempo e de recursos computacionais no treinamento de um modelo, já que o treinamento de um modelo partindo do zero exige uma quantidade muito maior de tempo, de uma infraestrutura mais poderosa e um número muito maior de amostras no *dataset* de treinamento.

O TOD fornece uma coleção de modelos pré treinados, chamada de *Model Zoo,* esse recurso objetiva facilitar o treinamento de novos modelos utilizando *transfer learning*

a partir dos modelos fornecidos que foram treinados utilizando técnicas e arquiteturas de redes conhecidas pelos seus resultados nas competições de detecção de objetos como: *Faster R-CNN* (REN et al., 2015), *SSD (Single Shot Multibox Detector)* (LIU et al., 2016), *Inception* (SZEGEDY et al., 2015) *e Mobilenet* (HOWARD et al., 2017).

Os modelos do *Model Zoo* foram treinados utilizando os *datasets* das principais competições de detecção de objetos, como *COCO Dataset (Commom Objects in Context)* (LIN et al., 2014), *Kitti Dataset* (GEIGER et al., 2013) e *Open Image Dataset* (KUZNETSOVA et al., 2018). Estes conjuntos de imagens são compostos por milhões de imagens, em diversos contextos diferentes e contendo objetos de centenas de classes diferentes. Estes modelos foram treinados utilizando infraestruturas muito caras e complexas nos *datacenters* do Google por centenas de horas, tais requisitos são impeditivos para aplicações de pequeno porte e/ou experimentais em razão de seu custo operacional, dessa forma a utilização do *transfer learning* em conjunto com os modelos fornecidos na coleção facilita muito o treinamento de novos modelos e a experimentação.

A coleção fornece 35 modelos pré-treinados, e apresenta como parâmetros de comparação, a velocidade no processamento de uma imagem de 600x600 pixels e a mAP alcançada, que é a métrica de desempenho padrão de publicações de detecção de objetos. Por razões de tempo e recursos disponíveis, não seria possível testar todos os modelos fornecidos, dessa forma, optou-se por utilizar dois modelos escolhidos com base em seus resultados nas principais competições de reconhecimento de objetos (ILSVRC 2014, ILSVRC 2015 (RUSSAKOVSKY et al., 2015) e COCO 2015 (LIN et al., 2014)), a experimentação com os demais modelos fornecidos pode ser objeto de um trabalho futuro. O SI2 (ssd_inception_v2_coco) utiliza uma combinação da rede SSD e *Inception V2* treinado sobre o *dataset* COCO, tendo conseguido 22 mAP e 31 milissegundos no treinamento prévio. O FRI2 (faster_rcnn_inception_resnet_v2_atrous_coco) utiliza uma combinação das redes *Faster RCNN* e *Inception V2* treinado sobre o *dataset* COCO tendo conseguido 28

mAP e 58 milissegundos no treinamento prévio.

Todo processo de treinamento e testes dos modelos, incluindo a definição do protocolo de avaliação, seguem os parâmetros setados em arquivos de configuração (*pipeline files*). O pacote que acompanha cada modelo também trás exemplos de arquivos de configuração para servir de base na configuração do processo de treinamento e testes. Os autores (PKULZC; RATHOD; WU, 2019) afirmam que os arquivos de configuração fornecidos são os mesmos usados para o pré-treinamento de cada modelo e fornecem um ponto de partida para o re-treinamento de novos modelos.

Neste trabalho, tentou-se utilizar os arquivos com o mínimo de alterações possíveis, a fim de perseguir a qualidade de detecção informada pela biblioteca, ajustando apenas parâmetros que adequassem o treinamento à tarefa e ao *hardware* disponível. A otimização destes parâmetros pode ser objeto de trabalhos futuros. Na figura 18 é possível ver parte do arquivo de configuração do modelo SI2.

```
model {
  ssd {
    num_classes: 1
    box_coder {
      faster_rcnn_box_coder {
        y_scale: 10.0
        x_scale: 10.0
        height_scale: 5.0
        width_scale: 5.0
      }
    }
    matcher {
      argmax_matcher {
        matched_threshold: 0.5
        unmatched_threshold: 0.5
        ignore_thresholds: false
        negatives_lower_than_unmatched: true
        force_match_for_each_row: true
      }
    }
    similarity_calculator {
      iou_similarity {
      }
    }
    anchor_generator {
      ssd_anchor_generator {
        num_layers: 6
        min_scale: 0.2
        max_scale: 0.95
        aspect_ratios: 1.0
        aspect_ratios: 2.0
        aspect_ratios: 0.5
        aspect_ratios: 3.0
        aspect_ratios: 0.3333
        reduce_boxes_in_lowest_layer: true
      }
    }
    image_resizer {
      fixed_shape_resizer {
        height: 300
        width: 300
```

*Figura 18: Trecho do Arquivo de Configuração da Pipeline do modelo SI2.*

Sobre os parâmetros utilizados, o único parâmetro ajustados foi o número de classes,

definido como 1, já que objetivava-se a detecção de apenas uma classe (glomérulos). A quantidade de ciclos de treinamento para os dois modelos foi 200.000 ciclos, a fim de comparar o tempo total gasto no treinamento. Por conta das diferenças entre os modelos, alguns parâmetros são diferentes entre si, ou seja, alguns parâmetros não estão presentes em ambos os arquivos de configuração. A documentação da ferramenta é incompleta e a interpretação dos argumentos contidos nos arquivos por vezes depende de dedução e testes. No Apêndice A e B os arquivos de configuração utilizados neste trabalho são apresentados na íntegra.

Alguns parâmetros que valem ser citados aqui são: para o SI2, todas as imagens de entrada são redimensionadas dinamicamente na entrada para 300x300 pixels durante a execução do treinamento, o tamanho do lote de treinamento foi de 32 amostras, o tamanho do lote está relacionado com o poder de processamento e memória disponíveis na máquina, este número foi adequado para a execução deste trabalho. Técnicas de *data augmentation* foram utilizadas, nesse caso o espelhamento horizontal e recorte aleatório. Como parâmetros para o FRI2, todas as imagens de entrada são redimensionadas para no mínimo 600 pixels e máximo 1024 pixels dinamicamente na entrada, durante a execução do treinamento. O tamanho do lote de treinamento é 1 amostra, no *hardware* disponível esse era o número máximo possível. A técnica de *data augmentation* utilizada é o espelhamento horizontal.

## 4.4 Treinamento dos Modelos

A partir da configuração da plataforma e da criação dos arquivos de configuração, o processo de treinamento e testes foi iniciado e monitorado utilizando a ferramenta *Tensorboard* que acompanha o *Tensorflow* e permite analisar o andamento do processo.

A medida que o treinamento foi avançando, testes de validação foram realizados a cada intervalo definido no *pipeline.* Após cada teste, gráficos de desempenho foram atualizados e através da ferramenta foi possível avaliar o andamento do treinamento.

O gráfico que exibe o mAP é o indicativo geral do desempenho do treinamento e espera-se ver uma curva ascendente que estabiliza a partir de certo ponto, o treinamento foi repetido seguindo a configuração do *pipeline,* porém pode ser interrompido a qualquer ciclo. O *framework* armazena uma quantidade configurável de cópias de segurança (*checkpoints*) das etapas de treinamento, em caso de falha, ou caso o desempenho do mAP caia ao invés de melhorar, o processo pode ser interrompido e os últimos *checkpoints* utilizados para exportar o modelo treinado no ponto desejado.

Durante o treinamento, o *framework* executou um teste a cada *checkpoint*, ou seja um a cada 1.000 ciclos, apesar da documentação apresentar um parâmetro para determinar o número máximo de testes durante o treinamento, que havia sido definido em 10 para ambos os treinamento, a documentação não explica o porque desse comportamento. Na Figura 19 pode-se ver o gráfico de mAP gerados pelo *Tensorboard* a medida que o processo de treinamento acontece, o eixo x traz o ciclo de treinamento e o eixo y o mAP conseguido naquele ciclo.

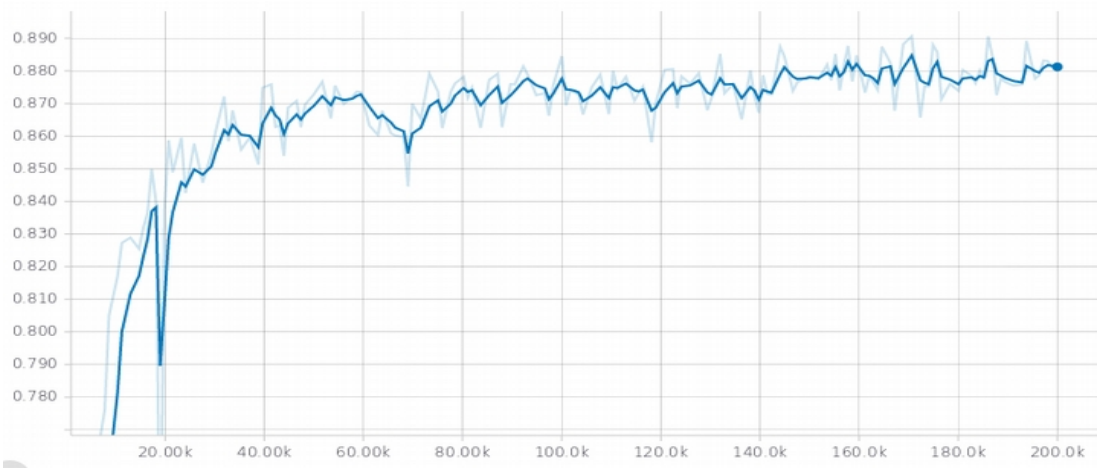



*Figura 19: Gráfico de mAP ao longo dos ciclos de treinamento. Eixo x ciclos de treinamento, eixo y o mAP alcançado no ciclo.*

De acordo com a documentação do *framework,* os *datasets* de treinamento e validação são independentes entre si, ou seja, as imagens do *dataset* de validação não são utilizadas no processo de treinamento, são utilizadas apenas para medir o desempenho do modelo a medida que o treinamento avança, dessa forma, já seria suficiente para

determinar o desempenho do modelo. Porém foram realizados testes com o *dataset* exclusivo para os testes finais, para averiguar se os números de desempenho se repetiam diante de imagens não fornecidas durante o treinamento.

Ao final do processo de treinamento uma nova execução foi realizada utilizando os mesmos arquivos de configuração, porém agora utilizando o *dataset* de testes a fim de medir o desempenho com imagens inéditas.

O tempo total de treinamento do modelo FRI2 foi 54 horas e 19 minutos e foi interrompido ao atingir 100.000 ciclos, metade do treinamento planejado, em razão do gráfico de desempenho indicar uma estabilização da curva desde o 55.000 e uma tendência de queda a partir desse ponto. A quantidade de *checkpoints*, por padrão, havia sido definido para armazenar os 5 últimos *checkpoints*, se o treinamento houvesse continuado e a tendência persistisse, não haveria possibilidade de recuperar o *checkpoint* onde houveram os melhores resultados. O modelo modelo SI2 levou 38 horas e 54 minutos, completando todo seu ciclo de treinamento planejado de 200.000 ciclos.

## 4.5 Protocolos de Avaliação do TOD

O *framework* suporta três protocolos de avaliação, os mesmos utilizados nas principais competições de detecção de objetos, o *PASCAL VOC*, *COCO* e *Open Images*, a escolha do protocolo é feita através do arquivo de configuração, o protocolo utilizado neste trabalho foi o *COCO,* já que os modelos utilizados foram treinados previamente utilizando o *dataset* da mesma competição.

Este protocolo traz como métricas de avaliação:

- mAP: com limiar de corte 0.5 à 0.95 com passo 0.05, uma média dos 10 valores no intervalo, ou seja, caixas com IoU abaixo ou acima desse limiar não são consideradas;
- mAP@0.5 e mAP@0.75: limiar de corte igual a 0.5 e 0.75 respectivamente;
- *AP Across Scales*: mAP por tamanho de objetos, *small* para $< 32^2$ (32x32)

pixels, *medium* para objetos $32^2 <$ área $< 96^2$ e *large* para objetos com área $> 96^2$;

- AR (*Average Recall*): $AR^{max=1}$, $AR^{max=10}$, $AR^{max=100}$ para detecções de objetos por imagem, respectivamente, máximo 1, 10 e 100 objetos;
- AR *Across Scales*: *Recall* máximo por tamanho de objetos, *small* para $< 32^2$ (32x32) pixels, *medium* para objetos $32^2 <$ área $< 96^2$ e *large* para objetos com área $> 96^2$;

Todas as métricas apresentadas no protocolo foram utilizadas neste trabalho. Além disso, como já discutido na Seção 2.2.6 (Análise de Desempenho), apesar do protocolo apresentado ser amplamente utilizado nas competições de detecção de objetos, diversos trabalhos apresentam seus resultados utilizando as métricas comuns em trabalhos de Aprendizagem de Máquina, são elas: *Precision*, *Recall* e *F1 Score. P*or este motivo, a fim de comparar os resultados deste trabalho com trabalhos relacionados, uma análise manual dos resultados dos testes finais foi realizada, ou seja, as imagens com o resultado da detecção sobre o *dataset* de testes finais foram analisadas, imagem por imagem. As caixas de fronteira detectadas foram contadas, colhendo os dados necessários para o cálculo das métricas *precision, recall* e *F1 Score*. Esta análise manual foi necessária porque o *framework* utilizado não permite ajustes no protocolo de métricas.

# **Capítulo 5** Resultados e Discussões

Neste capítulo serão apresentados e discutidos os resultados alcançados com a metodologia proposta por este trabalho.

Como já mencionado, o *framework* utilizado (TOD) requer a escolha do protocolo de avaliação a ser utilizado entre os protocolos disponíveis, o protocolo utilizado na competição COCO (LIN et al., 2014) foi o escolhido, este protocolo define as métricas que o sistema irá calcular durante o teste e que retornará no relatório de avaliação. A ferramenta não permite a configuração personalizada de métricas, somente as métricas aceitas na competição.

Ao final da etapa de treinamento dos modelos SI2 e FRI2, os dados foram coletados e são apresentados na Tabela 1. Os dados refletem o protocolo descrito na seção 4.5.

| | **SI2** | **FRI2** |
|---|---|---|
| **mAP** | 0,8812 | 0,8821 |
| **mAP@.50** | 0,9886 | 0,9983 |
| **mAP@.75** | 0,9704 | 0,9929 |
| **mAP$^{\text{Small}}$** | -1,0000 | -1,0000 |
| **mAP$^{\text{Medium}}$** | -1,0000 | -1,0000 |
| **mAP$^{\text{Large}}$** | 0,8812 | 0,8821 |
| **AR@1** | 0,7795 | 0,7812 |
| **AR@10** | 0,9113 | 0,9912 |
| **AR@100** | 0,9116 | 0,9200 |
| **AR@100$^{\text{large}}$** | 0,9116 | 0,9200 |
| **AR@100$^{\text{medium}}$** | -1,0000 | -1,0000 |
| **AR@100$^{\text{small}}$** | -1,0000 | -1,0000 |

*Tabela 1: Resultados do Treinamento.*

Após a realização dos testes finais, os dados foram coletados e são apresentados na Tabela 2.

| | **SI2** | **FRI2** |
|---|---|---|
| **mAP** | 0,8831 | 0,8723 |
| **mAP@.50** | 0,9971 | 0,9974 |
| **mAP@.75** | 0,9743 | 0,9918 |
| **mAP$^{\text{Small}}$** | -1,0000 | -1,0000 |
| **mAP$^{\text{Medium}}$** | -1,0000 | -1,0000 |
| **mAP$^{\text{Large}}$** | 0,8831 | 0,8724 |
| **AR@1** | 0,7744 | 0,7375 |
| **AR@10** | 0,9080 | 0,9113 |
| **AR@100** | 0,9088 | 0,9113 |
| **AR@100$^{\text{large}}$** | 0,9088 | 0,9113 |
| **AR@100$^{\text{medium}}$** | -1,0000 | -1,0000 |
| **AR@100$^{\text{small}}$** | -1,0000 | -1,0000 |

*Tabela 2: Resultados dos Testes Finais*

Da análise dos dados pode-se ver que entre os testes feitos durante o treinamento e os testes finais existe uma distância muito pequena, com resultados ligeiramente melhores nos testes finais, o que é um comportamento atípico, porém justificável levando em consideração essa pequena variação. Isso confirma que o *dataset* de testes do treinamento de fato não é utilizado para o aprendizado, mas somente para testes a fim de monitorar o progresso do treinamento, o que preserva a capacidade de generalização do modelo.

A análise dos testes finais mostra que o desempenho geral (mAP) é bem aproximado entre os dois modelos, com pouco mais de 0,01 ponto de distância entre eles. O SI2 (0,8831) é um pouco melhor que o FRI2 (0,8723) em mAP. Quando observa-se o mAP por corte de IoU (mAP@.50, mAP@.75. Quando a *boundary box* sobrepõe 50% ou 75% da área da caixa *ground-truth*, respectivamente), vê-se que os dois modelos tem praticamente o mesmo desempenho em mAP@.50 (SI2 – 0,9971 e FRI2 –

0,9974). Em mAP@.75 vê-se que o modelo FRI2 é um pouco melhor (SI2 – 0,9743 e FRI2 – 0,9918). O desempenho por escala ($mAP^{small}$, $mAP^{medium}$, $mAP^{large}$) indica que não houveram objetos *small* (<32x32 *pixels*) e *medium* (>32x32 & <96x96 *pixels*) etiquetados no *dataset*, desse modo, $mAP^{large}$(>96x96 *pixels*) trazem os mesmos números de mAP. Quando observa-se o *recall* por número de objetos (AR@1 [até 1 glomérulo por imagem], AR@10 [até 10], AR@100 [até 100]) vê-se que, em imagens de um único glomérulo, o SI2 (0,7744) é melhor que o FRI2(0,7375). Já em imagens com até 10 glomérulos o FRI2 (0,9113) se sai melhor que o SI2 (0,9080), nota-se também que ambos os modelos erram menos em imagens com mais de um glomérulo. Não houveram imagens com mais de 10 glomérulos, por isso AR@100 refletem os mesmos números de AR@10. Da mesma forma, o AR@100 divididos por escala ($AR@100^{large}$, $AR@100^{medium}$, $AR@100^{small}$) refletem o mesmo comportamento, já que só existiram imagens consideradas *large* e com até 10 glomérulos.

Apesar das semelhanças apresentadas, existe uma grande discrepância no tempo necessário para processamento de cada imagem. O FRI2 precisa de, em média, 30,91 segundos para processar uma imagem, enquanto o SI2 precisa somente de 0,79 segundo. Com isso, o SI2 é 30,81 vezes mais rápido que o FR2 no processamento de uma imagem. Essa diferença de tempo também foi observada durante o treinamento do modelo, como discutida na seção 4.4 (FRI2 54 horas e 19 minutos para 100.000 ciclos, enquanto SI2 levou 38 horas e 54 minutos para 200.000 ciclos).

O mAP é a métrica mais comum em problemas de detecção de objetos, é um número sobre o desempenho geral de um modelo, que vai de 0,0 a 1,0, sendo que o 1,0 representa o modelo perfeito, nota-se que os números alcançados pelos modelos estão próximos ao resultado ideal (SI2 – 0,8831 mAP; FRI2 - 0,8723 mAP).

Nota-se também que, quando comparados aos índices fornecidos pela biblioteca de modelos pré-treinados (*Model Zoo*)(SI2 - 0,22 mAP, FRI2 - 0,28 mAP), obviamente resguardadas as devidas proporções, já que o objetivo da competição COCO (LIN et al., 2014) é detectar centenas de classes diferentes, enquanto este trabalho objetiva a

detecção de apenas uma classe (glomérulos), conseguiu-se bons resultados.

Nas figuras a seguir é possível observar alguns resultados na detecção de glomérulos pelos modelos SI2 e FRI2. Note que em cada figura a imagem da esquerda refere-se a um resultado obtido pelo modelo FRI2 e a imagem da direita um resultado obtido pelo modelo SI2. Nestas figuras os retângulos com borda amarela representam as caixas de fronteira de referência (*ground-truth boundary boxes*) e os retângulos com borda verde representam as caixas de fronteira detectadas pelo respectivo modelo que devem conter um glomérulo. O retângulo verde também acompanha o número de confiança (*confidence*), uma percentagem que indica se o modelo teve dificuldades em rotular a caixa.

Na Figura 20 é possível observar o resultado obtido por ambos os modelos sobre uma imagem que contém um glomérulo saudável. Observa-se, neste exemplo, que os retângulos verde e amarelo estão sobrepostos, o que indica um IoU alto e que a detecção alcançou seu objetivo com ótima qualidade. A confiança para ambos os modelos é 100%, o que indica que não tiveram dificuldades em classificar as caixas. Nota-se ainda que, neste exemplo, os modelos apresentaram resultados quase idêntico.

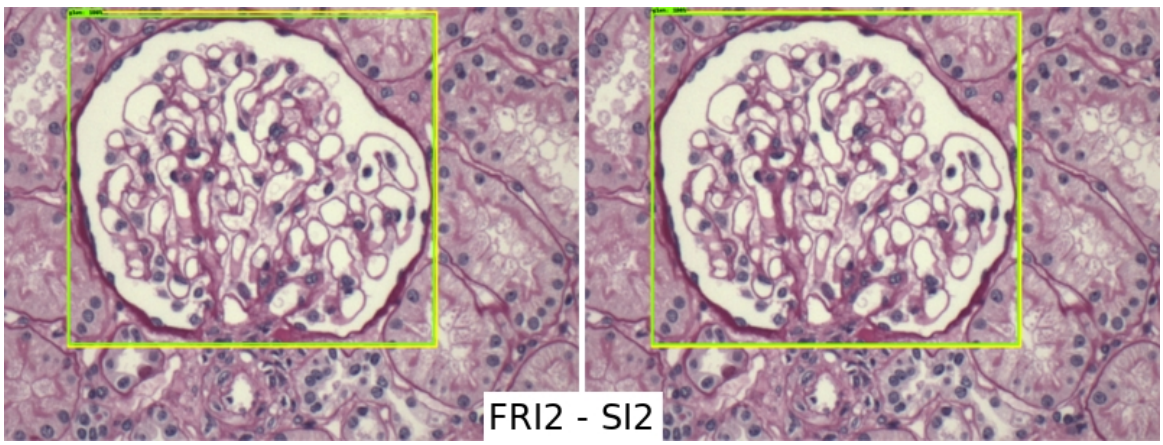



*Figura 20: Resultado da detecção de um glomérulo. Á esquerda do modelo FRI2 e à direita do modelo SI2.*

Nas Figuras 21 e 22 observa-se que os modelos conseguem detectar glomérulos mesmo em imagens cujos tecidos foram preparados com corantes diferentes. A Figura 21 traz um glomérulo acometido por glomerulopatia membranosa. Os resultados foram um pouco diferentes entre si, o modelo SI2 propôs uma caixa mais larga, mas ambos os

resultados englobam todo o glomérulo. A Figura 22 traz um glomérulo saudável. Nota-se que a imagem apresenta outro tipo de corante, além de trazer um glomérulo em uma escala de aproximação, ou um tamanho diferente. Esta característica dificulta a detecção de objetos e é comum no *dataset* utilizado neste trabalho.

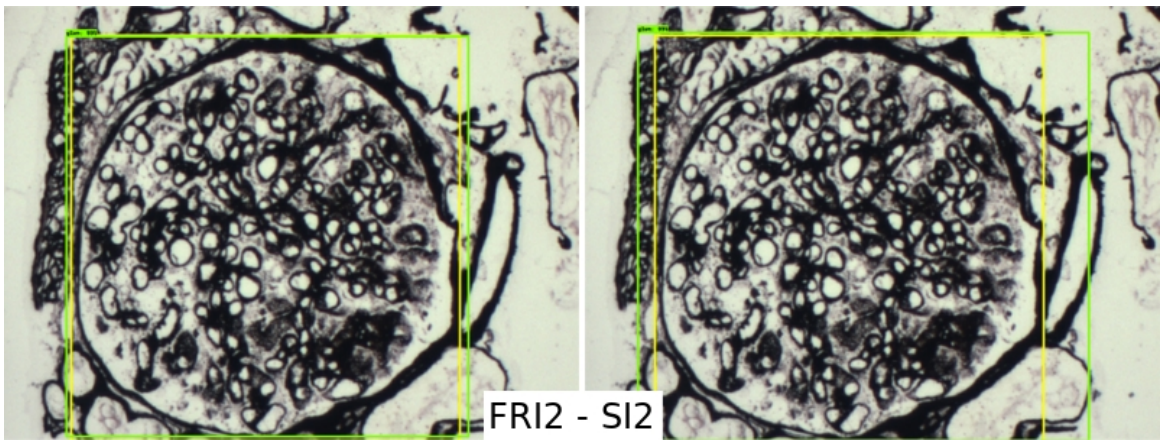



*Figura 21: Detecção de um glomérulo com diferentes corantes. FRI2 / SI2.*

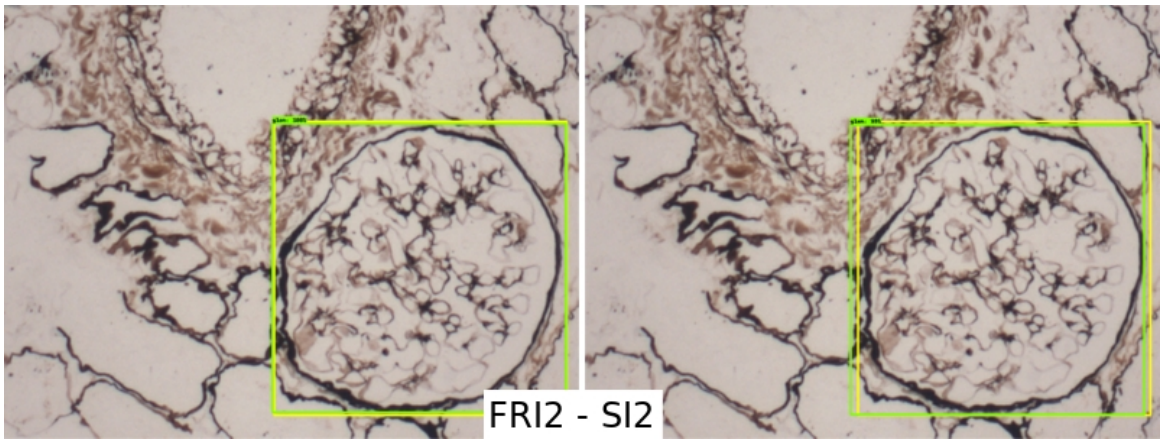



*Figura 22: Outro exemplo de detecção com variação no corante. FRI2 / SI2.*

A Figura 23 traz os resultados dos modelos sobre uma imagem que contém mais de um glomérulo, esta imagem contém quatro glomérulos acometidos por glomerulopatia membranosa.

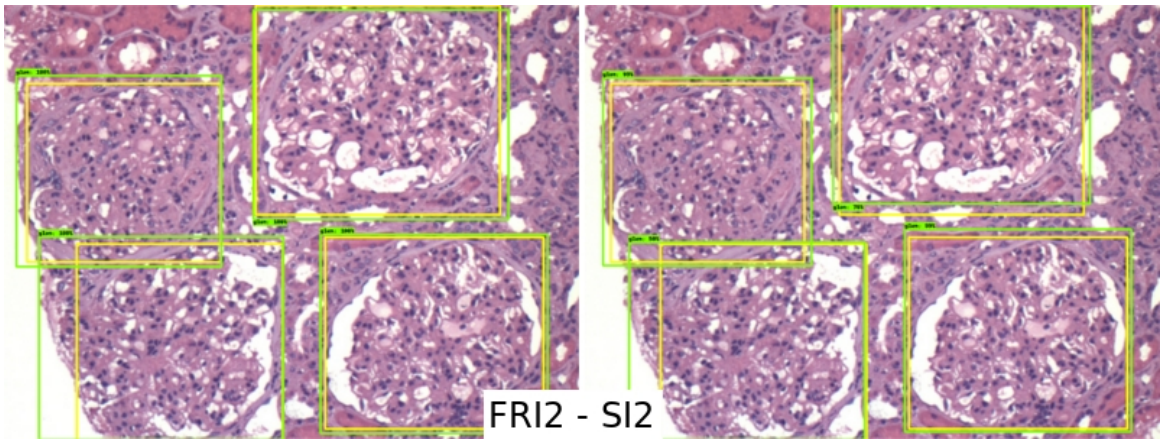



*Figura 23: Detecção em imagens com vários glomérulos. FRI2 / SI2.*

Na Figura 24 vê-se os resultados dos modelos sobre uma imagem com 3 glomérulos acometidos por glomerulopatia membranosa. Nesta figura observa-se a diferença entre o comportamento dos modelos, onde o modelo SI2 não consegue detectar os glomérulos na região periférica da imagem.

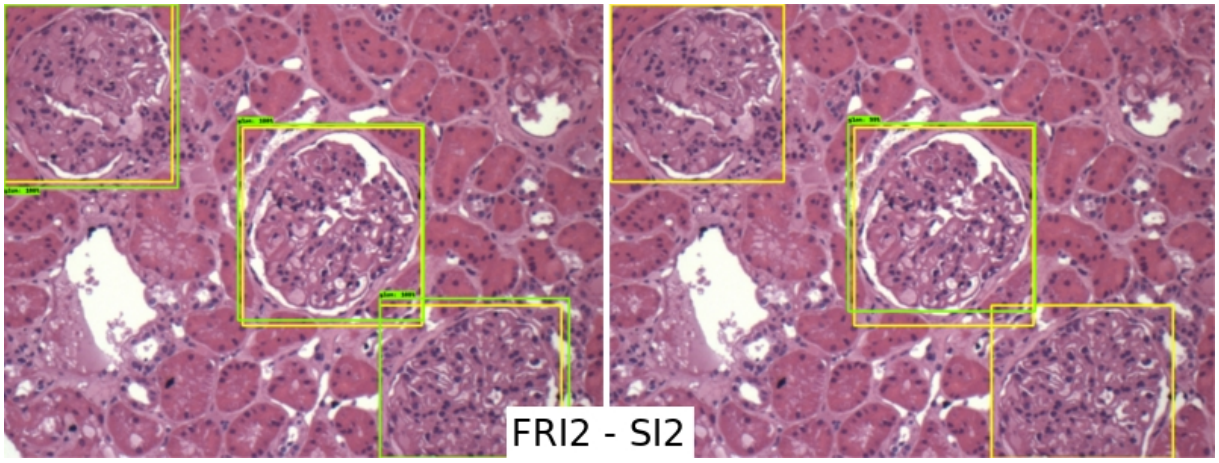



*Figura 24: Diferenças entre os modelos em imagens com vários glomérulos. FRI2 / SI2.*

A Figura 25 traz um exemplo de detecção sobre uma imagem onde os glomérulos ocupam uma área relativa menor que as imagens anteriores, indicando uma escala de aproximação menor e mostrando que, apesar disso, os modelos conseguiram obter sucesso. A figura traz glomérulos acometidos por glomerulopatia membranosa.

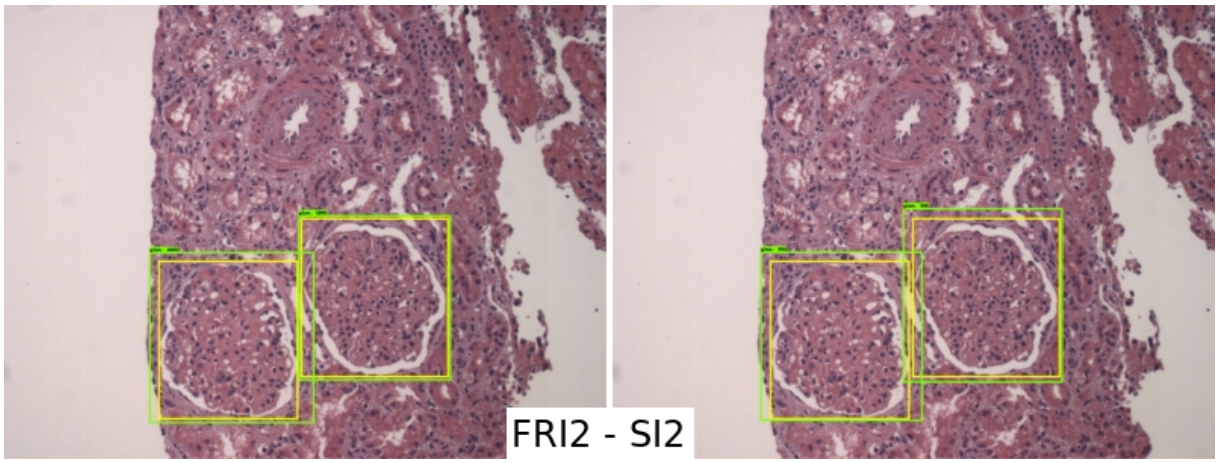



*Figura 25: Detecção em escalas de aproximação variadas. FRI2 / SI2.*

As Figuras 26 e 27 trazem exemplos de detecção em imagens que contém glomérulos doentes e uma variação expressiva em sua morfologia, apesar dessa condição os modelos conseguem detectar os glomérulos. A Figura 26 contém um glomérulo acometido por glomerulosclerose segmentar e a Figura 27 por glomerulopatia membranosa.

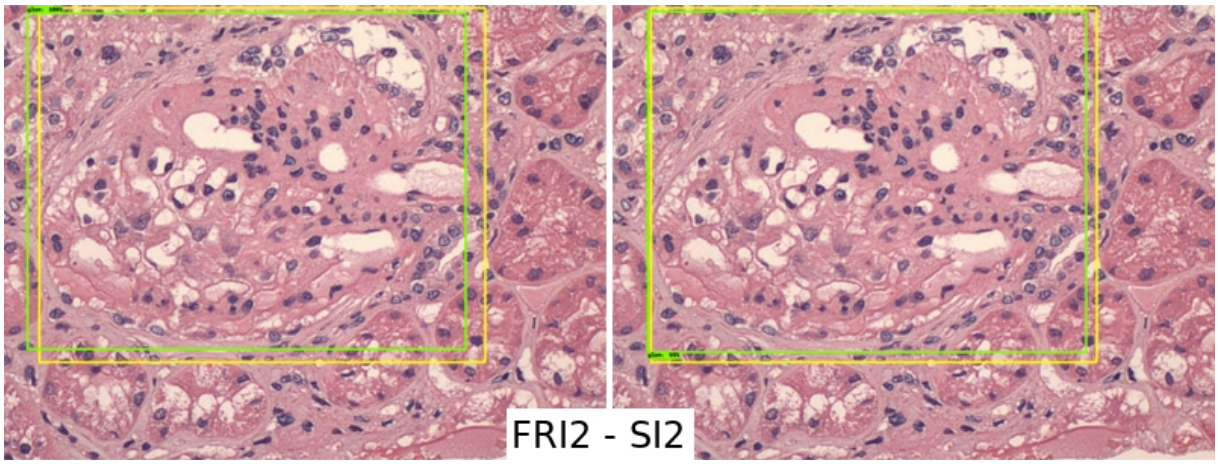



*Figura 26: Detecção em glomérulos com glomerulosclerose segmentar. FRI2 / SI2.*

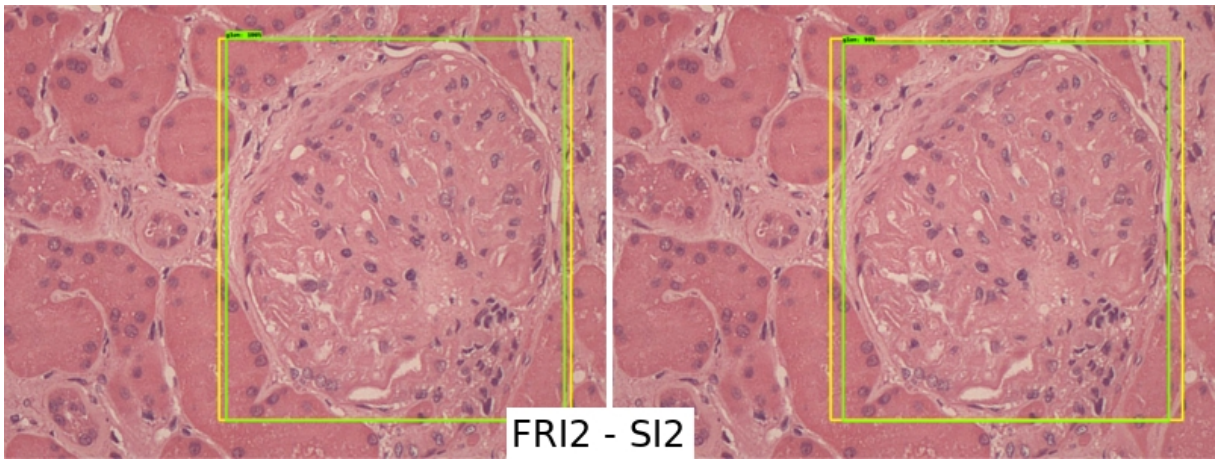



*Figura 27: Detecção em glomérulos com glomerulopatia membranosa. FRI2 / SI2.*

A comparação dos resultados alcançados por este trabalho com os trabalhos relacionados (Capítulo 3) é uma tarefa difícil, haja vista que, os trabalhos sobre esse tema são escassos, a maioria busca a segmentação e não a detecção. Gallego et al. (2018) e Simon et al. (2018) propõem a detecção de glomérulos, porém utilizam imagens de slide completa WSI ( *Whole Slide Image*), que são imagens capturadas por escaneamento, com escala de aproximação padronizada, já que contempla toda a lâmina física e esta tem tamanho padronizado. Enquanto este trabalho usa imagens em escalas de aproximação incerta, capturadas por fotografia digital. Além disso, os referidos trabalhos não apresentam seus resultados nas métricas comuns ao problema de detecção de objetos, o mAP. Apesar disso, para fins de comparação, uma contagem manual foi feita a fim de calcular as métricas utilizadas pelos trabalhos referidos. Isso foi necessário porque o *framework* não fornece a possibilidade de personalização do protocolo de avaliação.

A Tabela 3 apresenta os valores de *recall*, *precision* e *F1-Score* conseguidos pelos modelos. Pode-se observar que o modelo SI2 é melhor em *precision*, o que indica uma menor ocorrência de falsos positivos, ou seja, quando o modelo indica caixas onde não existem glomérulos. Já em *recall*, o modelo FRI2 se saiu melhor, indicando menos ocorrências de falsos negativos, que ocorrem quando o modelo não indica caixas onde de fato existiam glomérulos. A métrica *F1 Score* é a média harmônica entre *recall* e *precision,* objetiva representar um índice geral de desempenho do modelo. Vê-se que o modelo FRI2 é o modelo mais equilibrado dentre ambos.

| | *Precision* | *Recall* | *F1 Score* |
|---|---|---|---|
| **SI2** | 0,99 | 0,90 | 0,94 |
| **FRI2** | 0,94 | 0,99 | 0,97 |

*Tabela 3: Precision, Recall e F1-Score*

A Tabela 4 traz o desempenho dos modelos, porém considerando as glomerulopatias de forma separada. A coluna modelo indica o agrupamento de medidas para cada um dos modelos, a coluna condição indica a condição do glomérulo considerada para os dados de cada linha, que podem apresentar-se como sadio, GS (glomerulosclerose segmentar) e GM (glomerulopatia membranosa). As demais colunas trazem os dados de desempenho, a exemplo das tabelas anteriores.

Analisando os dados da Tabela 4, nota-se que o modelo SI2 tem mais dificuldade na detecção de glomérulos saudáveis, com *F1 Score* de 0,92, *recall* de 0,87 indicando ocorrência de casos em que glomérulos não foram detectados. O modelo SI2 consegue um melhor desempenho com glomérulos acometidos por glomerulopatia membranosa, com *F1 Score* 0,98. Em contrapartida o modelo FRI2 tem seu pior desempenho com glomérulos acometidos por glomerulopatia membranosa, com *F1 Score* 0,93, *precision* 0,87 indica ocorrência de casos em que outras estruturas foram detectadas como se fossem glomérulos. Seu melhor desempenho foi com glomérulos saudáveis, com *F1 Score* 0,98.

| Modelo | Condição | *Precision* | *Recall* | *F1 Score* |
|---|---|---|---|---|
| **SI2** | Sadios | 0,99 | 0,87 | 0,92 |
| | GS | 1,00 | 0,96 | 0,98 |
| | GM | 1,00 | 0,89 | 0,94 |
| **FRI2** | Sadios | 0,97 | 0,99 | 0,98 |
| | GS | 0,87 | 1,00 | 0,93 |
| | GM | 0,96 | 1,00 | 0,98 |

*Tabela 4: Desempenho por condição do glomérulo.*

A Tabela 5 apresenta uma comparação do desempenho alcançado por este trabalho e trabalhos relacionados a ele, destacando que uma comparação direta não é possível, como já discutido anteriormente. No trabalho de Gallego et al. (2018), os autores conseguiram 0,88 de *precision*, 1,0 de *recall* e 0,937 de *F1 Score*. Já Simon et al. (2018) alcançaram 0,92 de *precision*, e 0,76 de *recall e 0,83 de F1 Score* para imagens contendo glomérulos saudáveis e 0,90 de *precision*, e 0,77 de *recall e 0,83 de F1 Score* para imagens contendo glomérulos doentes. Pode-se ver que este trabalho traz desempenho similar ao de Gallego et al. (2018) e Simon et al. (2018), não sendo possível afirmar que é melhor devido as diferenças entre eles, o que impede a comparação direta.

| | *Precision* | *Recall* | *F1 Score* |
|---|---|---|---|
| SI2 | **0,99** | 0,90 | 0,94 |
| FRI2 | 0,94 | 0,99 | **0,97** |
| Gallego et al. (2018) | 0,88 | **1,00** | 0,94 |
| Simon et al. (2018) - Glomérulos Sadios | 0,92 | 0,76 | 0,83 |
| Simon et al. (2018) - Glomérulos Doentes | 0,90 | 0,77 | 0,83 |

*Tabela 5: Comparação entre trabalhos.*

# **Capítulo 6** Considerações Finais

Este trabalho teve como objetivo propor uma metodologia capaz de detectar automaticamente glomérulos em imagens histológicas digitalizadas, a fim de aprimorar as capacidades do sistema PathoSpotter.

Para alcançar estes objetivos, técnicas de *Deep Learning* foram utilizadas através do *framework* Tensorflow, os modelos foram treinados usando uma máquina virtual em um *datacenter* remoto. *Datasets* foram criados usando imagens histológicas digitais capturadas por fotografia digital, que continham glomérulos e outras estruturas em diversas escalas. As imagens continham glomérulos sadios e glomérulos acometidos por glomerulopatias (glomerulopatia membranosa e glomerulosclerose segmentar). Os modelos pré-treinados SI2 e FRI2 foram retreinados e testados usando um *dataset* próprio para testes e os resultados foram aferidos e coletados. Utilizando a métrica padrão de trabalhos de detecção de objetos, nosso melhor resultado foi mAP de 0,8831 para o modelo SI2, contra 0,8723 de FRI2.

Ao analisar as métricas clássicas de aprendizagem de máquina, observou-se que o modelo FRI2 obteve 0,97 de *F1 Score*, contra 0,94 do modelo SI2. Observa-se que o modelo SI2 foi o melhor em mAP enquanto FRI2 foi melhor em *F1 Score*, nota-se também que, a diferença no desempenho geral de ambos é bem pequena (0,0108 mAP e 0,03 *F1 Score*). Dessa forma, esta alternância pode ser atribuída às diferentes metodologias de levantamento dos dados, ou ainda ao sorteio aleatório de imagens para os testes dos modelos, visto que, é natural uma variação pequena nos números a cada processo de testes. Quando o desempenho é analisado separando pelas glomerulopatias, conclui-se que o modelo SI2 tem mais dificuldade na detecção de glomérulos saudáveis (0,92 *F1 Score) e* consegue um melhor desempenho com

glomérulos acometidos por glomerulopatia membranosa (0,98 *F1 Score*). Já o modelo FRI2 tem mais dificuldade com glomérulos acometidos por glomerulopatia membranosa (0,93 *F1 Score*) e melhor desempenho com glomérulos saudáveis (0,98 *F1 Score*). A maior diferença entre eles se deu no tempo necessário para o treinamento e processamento de cada imagem. No treinamento, o modelo SI2 foi 64% mais rápido que FRI2 e no processamento de cada imagem, o modelo SI2 foi 98% mais rápido que FRI2. Diante da grande diferença dos tempos de execução e treinamento, e como o tempo é um fator determinante para cumprir o objetivo desta tarefa, a utilização do modelo SI2 é a mais adequada. Destaque-se que o desempenho aferido por este trabalho limita-se à detecção de glomérulos sadios e acometidos por glomerulopatia membranosa e glomerulosclerose.

As contribuições alcançadas por este trabalho indicam que a utilização desta metodologia pode ser eficaz na resolução do problema perseguido, contribuindo para o desenvolvimento de técnicas que tornem possível a detecção automática de glomérulos em imagens histológicas renais.

## 6.1 Pesquisas Futuras

Propõe-se na continuidade deste trabalho a otimização dos parâmetros no treinamento dos modelos utilizados. A utilização de imagens de glomérulos afetados por outras glomerulopatias não contempladas neste trabalho. O teste de outras arquiteturas de rede fornecidas pela biblioteca *Model Zoo* do *framework* TOD. Propõe-se também a adaptação do método proposto por este trabalho no processamento de imagens de *slide* completo (WSI). Propõe-se ainda o estudo da viabilidade da aplicação do modelo SI2, ou outra arquitetura de rede do tipo M*obilenet,* em arquiteturas móveis (*smartphones*), o que pode ser útil na detecção de glomérulos no ato da captura da imagem, com o acoplamento destes dispositivos diretamente ao microscópio óptico.

# Referências Bibliográficas

ABADI, M. et al. **TensorFlow: Large-scale machine learning on heterogeneous systems**. . In: 12TH USENIX SYMPOSIUM ON OPERATING SYSTEMS DESIGN AND IMPLEMENTATION. 2015Disponível em: <www.tensorflow.org>

BACKES, A. R.; SÁ JUNIOR, J. J. DE M. **Introdução à Visão Computacional Usando MATLAB**. [s.l.] Alta Books Editora, 2019.

BARROS, G. O. PathoSpotter: um sistema para classificação de glomerulopatias a partir de imagens histológicas renais. 2016.

BARROS, G. O. et al. PathoSpotter-K: A computational tool for the automatic identification of glomerular lesions in histological images of kidneys. **Scientific reports**, v. 7, p. 46769, 2017.

BARROS, G. O.; DOS-SANTOS, W. L. PathoSpotter: Um Sistema para Classificação de Glomerulopatias a partir de Imagens Histológicas Renais. **SIBGRAPI 2015**, 2015.

BELSARE, A. D.; MUSHRIF, M. M. Histopathological image analysis using image processing techniques: An overview. **Signal & Image Processing**, v. 3, n. 4, p. 23, 2012.

CHOLLET, F. **Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek**. [s.l.] MITP-Verlags GmbH & Co. KG, 2017.

CURRY, B. **An Introduction to Transfer Learning in Machine Learning.** Disponível em: <https://medium.com/kansas-city-machine-learning-artificial-intelligen/an-introduction-to-transfer-learning-in-machine-learning-7efd104b6026>. Acesso em: 25 jul. 2019.

DE ARAUJO, I. C.; SCHNITMAN, L.; DUARTE, A. A. Automated Detection of Segmental Glomerulosclerosis in Kidney Histopathology. **XIII Brazilian Congress on Computational Intelligence**, p. 12, 2017.

**Deep Learning Book**. [s.l.] Data Science Academy, 2018.

DESHPANDE, A. **A Beginner's Guide To Understanding Convolutional Neural Networks**. Disponível em: <https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/>. Acesso em: 25 jul. 2019.

EVERINGHAM, M. et al. The PASCAL Visual Object Classes Challenge: A Retrospective. **International Journal of Computer Vision (IJCV)**, v. 111, p. 98–136, 2015.

GALLEGO, J. et al. Glomerulus classification and detection based on convolutional neural networks. **Journal of Imaging**, v. 4, n. 1, p. 20, 2018.

GANDHI, R. **R-CNN, Fast R-CNN, Faster R-CNN, YOLO. Object Detection Algorithms**. Disponível em: <https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e>. Acesso em: 25 jul. 2019.

GARTNER, L. P.; HIATT, J. L. **Tratado de Histologia em Cores**. 3. ed. [s.l.] Elsevier, 2007.

GEIGER, A. et al. Vision meets robotics: The KITTI dataset. **The International Journal of Robotics Research**, v. 32, n. 11, p. 1231–1237, 2013.

GÉRON, A. **Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems**. [s.l.] O'Reilly Media, Inc., 2017.

GINLEY, B. et al. Unsupervised labeling of glomerular boundaries using Gabor filters and statistical testing in renal histology. **Journal of Medical Imaging**, v. 4, n. 2, p. 021102, 2017.

GINLEY, B.; TOMASZEWSKI, J. E.; SARDER, P. **Automatic computational labeling of glomerular textural boundaries**. Medical Imaging 2017: Digital Pathology. **Anais**...International Society for Optics and Photonics, 2017

GIRSHICK, R. et al. **Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation**. . In: PROCEEDINGS OF THE IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION. 2014Disponível em: <http://openaccess.thecvf.com/content_cvpr_2014/html/Girshick_Rich_Feature_Hierarchies_2014_CVPR_paper.html>. Acesso em: 25 jul. 2019

GIRSHICK, R. **Fast R-CNN**. The IEEE International Conference on Computer Vision (ICCV). **Anais**...dez. 2015Disponível em: <http://openaccess.thecvf.com/content_iccv_2015/html/Girshick_Fast_R-CNN_ICCV_2015_paper.html>

**Glomerulopatias**. Disponível em: <https://sbn.org.br/publico/doencas-comuns/glomerulopatias/>. Acesso em: 25 jul. 2019.

GONZALEZ, R. C.; WOODS, R. C. **Processamento Digital de Imagens**. [s.l.]

Pearson, 2009.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [s.l.] MIT Press, 2016.

HAM, J.; KAMBER, M.; PEI, J. **Data Mining: Concepts and Techniques**. 3. ed. [s.l.] Morgan Kaufmann, 2011.

HE, K. et al. Deep Residual Learning for Image Recognition. **arXiv:1512.03385 [cs]**, 10 dez. 2015.

HOSCH, W. L. **Machine Learning**, 2016. (Nota técnica).

HOWARD, A. G. et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. **arXiv:1704.04861 [cs]**, 16 abr. 2017.

HUANG, J. et al. **Speed/accuracy trade-offs for modern convolutional object detectors**. Disponível em: <https://github.com/tensorflow/models/tree/master/research/object_detection>.

HULSTAERT, L. **A Beginner's Guide to Object Detection**. Disponível em: <https://www.datacamp.com/community/tutorials/object-detection-guide>. Acesso em: 25 jul. 2019.

JUNQUEIRA, L. C.; CARNEIRO, J. **Histologia Básica**. 10. ed. [s.l.] Guanabara Koogan, 2013.

KARAGIANNAKOS, S. **Localization and Object Detection with Deep Learning**. Disponível em: <https://sergioskar.github.io/Localization_and_Object_Detection/>.

KARPATHY, A.; JOHNSON, J. **Transfer Learning**. Disponível em: <http://cs231n.github.io/transfer-learning/>.

KATO, T. et al. Segmental HOG: new descriptor for glomerulus detection in kidney microscopy image. **BMC bioinformatics**, v. 16, n. 1, p. 316, 2015.

KELLY, C.; LANDMAN, J. Anatomia do trato urinário. **Coleção Netter de ilustrações médicas. 2a ed. Rio de Janeiro: Saunders-Elsevier**, p. 24–7, 2014.

KUZNETSOVA, A. et al. The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale. **arXiv:1811.00982 [cs]**, 2 nov. 2018.

LIN, T. Y. et al. Microsoft COCO: Common Objects in Context. **Lecture Notes in Computer Science**, v. 8693, p. 740–755, 2014.

LIU, W. et al. SSD: Single Shot MultiBox Detector. **arXiv:1512.02325 [cs]**, v. 9905, p. 21–37, 2016.

MARÉE, R. et al. **An approach for detection of glomeruli in multisite digital pathology**. 2016 Ieee 13th International Symposium on Biomedical Imaging (Isbi). **Anais**...IEEE, 2016

MCCARTHY, N.; CUNNINGHAM, P.; OHURLEY, G. **The contribution of morphological features in the classification of prostate carcinoma in digital pathology images**. 2014 22nd International Conference on Pattern Recognition. **Anais**...IEEE, 2014

MELDAU, D. C. **Rim - Anatomia dos Rins - Sistema Excretor**. Disponível em: <https://www.infoescola.com/sistema-urinario/rim/>. Acesso em: 23 jul. 2019.

PARMAR, R. **Detection and Segmentation through ConvNets**. Disponível em: <https://towardsdatascience.com/detection-and-segmentation-through-convnets-47aa42de27ea>.

PEIXEIRO, M. **Hitchhiker's Guide to Residual Networks (ResNet) in Keras**. Disponível em: <https://towardsdatascience.com/hitchhikers-guide-to-residual-networks-resnet-in-keras-385ec01ec8ff>. Acesso em: 25 jul. 2019.

PETAR, V. **2D Convolution**. Disponível em: <https://github.com/PetarV-/TikZ/tree/master/2D%20Convolution>.

PKULZC; RATHOD, V.; WU, N. **Tensorflow detection model zoo**. Disponível em: <https://github.com/tensorflow/models>. Acesso em: 26 jul. 2019.

PONTI, M. A.; DA COSTA, G. B. Como funciona o Deep Learning. 2018.

PRAKASH, J. **Understanding and Implementing Architectures of ResNet and ResNeXt for state-of-the-art Image Classification: From Microsoft to Facebook [Part 1]**. Disponível em: <https://medium.com/@14prakash/understanding-and-implementing-architectures-of-resnet-and-resnext-for-state-of-the-art-image-cf51669e1624>. Acesso em: 25 jul. 2019.

PRATT, L. Y.; THRUN, S. **Machine Learning**. [s.l.] Springer, 1997. v. 28

RAJ, B. **A Simple Guide to the Versions of the Inception Network**. Disponível em: <https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202>. Acesso em: 25 jul. 2019.

REDMON, J. et al. **You Only Look Once: Unified, Real-Time Object Detection**. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). **Anais**...jun. 2016Disponível em: <https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Redmon_You_Only_Look_CVPR_2016_paper.html>

REN, S. et al. Faster R-CNN: Towards Real-Time Object Detection with Region

Proposal Networks. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 39, n. 6, p. 1137–1149, jun. 2017.

ROSSUM, G. V. Python tutorial, Technical Report. **Centrum voor Wiskunde en Informatica (CWI)**, 1995.

RUSSAKOVSKY, O. et al. ImageNet Large Scale Visual Recognition Challenge. **International Journal of Computer Vision (IJCV)**, v. 115, n. 3, p. 211–252, 2015.

SARAN, R. et al. US Renal Data System 2016 Annual Data Report: Epidemiology of Kidney Disease in the United States. **American Journal of Kidney Diseases**, v. 69, n. 3, p. A7–A8, 1 mar. 2017.

SARDER, P.; GINLEY, B.; TOMASZEWSKI, J. E. **Automated renal histopathology: Digital extraction and quantification of renal pathology**. . In: MEDICAL IMAGING 2016: DIGITAL PATHOLOGY. International Society for Optics and Photonics, 2016

SESSO, R. C. et al. Brazilian Chronic Dialysis Survey 2016. **Brazilian Journal of Nephrology**, v. 39, n. 3, p. 261–266, set. 2017.

SHAIK, F. **Understanding Inception Network from Scratch (with Python codes)**. Disponível em: <https://www.analyticsvidhya.com/blog/2018/10/understanding-inception-network-from-scratch/>. Acesso em: 25 jul. 2019.

SIMON, O. et al. Multi-radial LBP features as a tool for rapid glomerular detection and assessment in whole slide histopathology images. **Scientific reports**, v. 8, n. 1, p. 2032, 2018.

**Sistema Urinário**. Disponível em: <https://www.auladeanatomia.com/novosite/sistemas/sistema-urinario/>. Acesso em: 25 jul. 2019.

SOEIRO, E. M. D.; HELOU, C. M. DE B. Clinical, pathophysiological and genetic aspects of inherited tubular disorders in childhood. **Brazilian Journal of Nephrology**, v. 37, n. 3, p. 385–398, 2015.

SZEGEDY, C. et al. Going Deeper with Convolutions. **arXiv:1409.4842 [cs]**, 16 set. 2014.

SZEGEDY, C. et al. Rethinking the Inception Architecture for Computer Vision. **arXiv:1512.00567 [cs]**, 1 dez. 2015.

THOMÉ, F. S. et al. Brazilian chronic dialysis survey 2017. **Brazilian Journal of Nephrology**, v. 41, n. 2, p. 208–214, jun. 2019.

TZUTALIN. **LabelImg**. Disponível em: <https://github.com/tzutalin/labelImg>. Acesso em: 25 jul. 2019.

UNIVERSITY OF UTAH. **Renal Pathology**. Disponível em: <https://webpath.med.utah.edu/RENAHTML/RENAL080.html>. Acesso em: 25 jul. 2019.

VERONESE, F. J. V. et al. Síndrome Nefrótica Primária em adultos. 2010.

WAN, T. et al. **Wavelet-based statistical features for distinguishing mitotic and non-mitotic cells in breast cancer histopathology**. 2014 IEEE International conference on image processing (ICIP). **Anais**...IEEE, 2014

WAN, T. et al. Automated grading of breast cancer histopathology using cascaded ensemble with combination of multi-level image features. **Neurocomputing**, v. 229, p. 34–44, 2017.

YANN, L.; YOSHUA, B.; GEOFFREY, H. Deep learning. **Nature**, v. 521, p. 436–444, 28 maio 2015.

ZHANG, J.; HU, J.; ZHU, H. Contour extraction of glomeruli by using genetic algorithm for edge patching. **IEEJ transactions on electrical and electronic engineering**, v. 6, n. 3, p. 229–235, 2011.

ZHANG NON-MEMBER, J.; HU MEMBER, J.; ZHU NON-MEMBER, H. Contour Extraction of Glomeruli by Using Genetic Algorithm for Edge Patching. **IEEJ Transactions on Electrical and Electronic Engineering**, v. 6, p. 229–235, 2011.

# Apêndice A – Pipeline File FRI2

```
model {
  faster_rcnn {
    num_classes: 1
    image_resizer {
      keep_aspect_ratio_resizer {
        min_dimension: 600
        max_dimension: 1024
      }
    }
    feature_extractor {
      type: "faster_rcnn_inception_resnet_v2"
      first_stage_features_stride: 8
    }
    first_stage_anchor_generator {
      grid_anchor_generator {
        height_stride: 8
        width_stride: 8
        scales: 0.25
        scales: 0.5
        scales: 1.0
        scales: 2.0
        aspect_ratios: 0.5
        aspect_ratios: 1.0
        aspect_ratios: 2.0
      }
    }
    first_stage_atrous_rate: 2
    first_stage_box_predictor_conv_hyperparams {
      op: CONV
      regularizer {
        l2_regularizer {
          weight: 0.0
        }
```

```
    }
    initializer {
      truncated_normal_initializer {
        stddev: 0.00999999977648
      }
    }
  }
  first_stage_nms_score_threshold: 0.0
  first_stage_nms_iou_threshold: 0.699999988079
  first_stage_max_proposals: 300
  first_stage_localization_loss_weight: 2.0
  first_stage_objectness_loss_weight: 1.0
  initial_crop_size: 17
  maxpool_kernel_size: 1
  maxpool_stride: 1
  second_stage_box_predictor {
    mask_rcnn_box_predictor {
      fc_hyperparams {
        op: FC
        regularizer {
          l2_regularizer {
            weight: 0.0
          }
        }
        initializer {
          variance_scaling_initializer {
            factor: 1.0
            uniform: true
            mode: FAN_AVG
          }
        }
      }
      use_dropout: false
      dropout_keep_probability: 1.0
    }
  }
```

```
    second_stage_post_processing {
      batch_non_max_suppression {
        score_threshold: 0.0
        iou_threshold: 0.600000023842
        max_detections_per_class: 100
        max_total_detections: 100
      }
      score_converter: SOFTMAX
    }
    second_stage_localization_loss_weight: 2.0
    second_stage_classification_loss_weight: 1.0
  }
}
train_config {
  batch_size: 1
  data_augmentation_options {
    random_horizontal_flip {
    }
  }
  optimizer {
    momentum_optimizer {
      learning_rate {
        manual_step_learning_rate {
          initial_learning_rate: 0.000300000014249
          schedule {
            step: 900000
            learning_rate: 2.99999992421e-05
          }
          schedule {
            step: 1200000
            learning_rate: 3.00000010611e-06
          }
        }
      }
      momentum_optimizer_value: 0.899999976158
    }
```

```
    use_moving_average: false
  }
  gradient_clipping_by_norm: 10.0
  fine_tune_checkpoint:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/faster_rcnn_inception_resnet_v2_atrous_coco_2018_01_28/
model.ckpt"
  from_detection_checkpoint: true
  num_steps: 200000
}
train_input_reader {
  label_map_path:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/data/object-detection.pbtxt"
  tf_record_input_reader {
    input_path:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/data/train.record"
  }
}
eval_config {
  num_examples: 8000
  max_evals: 10
  use_moving_averages: false
}
eval_input_reader {
  label_map_path:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/data/object-detection.pbtxt"
  shuffle: false
  num_readers: 1
  tf_record_input_reader {
    input_path:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/data/test.record"
  }
}
```

# Apêndice B – *Pipeline File* SI2

```
model {
  ssd {
    num_classes: 1
    image_resizer {
      fixed_shape_resizer {
        height: 300
        width: 300
      }
    }
    feature_extractor {
      type: "ssd_inception_v2"
      depth_multiplier: 1.0
      min_depth: 16
      conv_hyperparams {
        regularizer {
          l2_regularizer {
            weight: 3.99999989895e-05
          }
        }
        initializer {
          truncated_normal_initializer {
            mean: 0.0
            stddev: 0.0299999993294
          }
        }
        activation: RELU_6
        batch_norm {
          decay: 0.999700009823
          center: true
          scale: true
          epsilon: 0.0010000000475
          train: true
        }
```

```
  }
  override_base_feature_extractor_hyperparams: true
}
box_coder {
  faster_rcnn_box_coder {
    y_scale: 10.0
    x_scale: 10.0
    height_scale: 5.0
    width_scale: 5.0
  }
}
matcher {
  argmax_matcher {
    matched_threshold: 0.5
    unmatched_threshold: 0.5
    ignore_thresholds: false
    negatives_lower_than_unmatched: true
    force_match_for_each_row: true
  }
}
similarity_calculator {
  iou_similarity {
  }
}
box_predictor {
  convolutional_box_predictor {
    conv_hyperparams {
      regularizer {
        l2_regularizer {
          weight: 3.99999989895e-05
        }
      }
      initializer {
        truncated_normal_initializer {
          mean: 0.0
          stddev: 0.0299999993294
```

```
        }
      }
      activation: RELU_6
    }
    min_depth: 0
    max_depth: 0
    num_layers_before_predictor: 0
    use_dropout: false
    dropout_keep_probability: 0.800000011921
    kernel_size: 3
    box_code_size: 4
    apply_sigmoid_to_scores: false
  }
}
anchor_generator {
  ssd_anchor_generator {
    num_layers: 6
    min_scale: 0.20000000298
    max_scale: 0.949999988079
    aspect_ratios: 1.0
    aspect_ratios: 2.0
    aspect_ratios: 0.5
    aspect_ratios: 3.0
    aspect_ratios: 0.333299994469
    reduce_boxes_in_lowest_layer: true
  }
}
post_processing {
  batch_non_max_suppression {
    score_threshold: 9.99999993923e-09
    iou_threshold: 0.600000023842
    max_detections_per_class: 100
    max_total_detections: 100
  }
  score_converter: SIGMOID
}
```

```
    normalize_loss_by_num_matches: true
    loss {
      localization_loss {
        weighted_smooth_l1 {
        }
      }
      classification_loss {
        weighted_sigmoid {
        }
      }
      hard_example_miner {
        num_hard_examples: 3000
        iou_threshold: 0.990000009537
        loss_type: CLASSIFICATION
        max_negatives_per_positive: 3
        min_negatives_per_image: 0
      }
      classification_weight: 1.0
      localization_weight: 1.0
    }
  }
}
train_config {
  batch_size: 24
  data_augmentation_options {
    random_horizontal_flip {
    }
  }
  data_augmentation_options {
    ssd_random_crop {
    }
  }
  optimizer {
    rms_prop_optimizer {
      learning_rate {
        exponential_decay_learning_rate {
```

```
          initial_learning_rate: 0.00400000018999
          decay_steps: 800720
          decay_factor: 0.949999988079
        }
      }
      momentum_optimizer_value: 0.899999976158
      decay: 0.899999976158
      epsilon: 1.0
    }
  }
  fine_tune_checkpoint:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/ssd_inception_v2_coco_2018_01_28/model.ckpt"
  from_detection_checkpoint: true
  num_steps: 200000
}
train_input_reader {
  label_map_path:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/data/object-detection.pbtxt"
  tf_record_input_reader {
    input_path:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/data/train.record"
  }
}
eval_config {
  num_examples: 8000
  max_evals: 10
  use_moving_averages: false
}
eval_input_reader {
  label_map_path:
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/data/object-detection.pbtxt"
  shuffle: false
  num_readers: 1
  tf_record_input_reader {
    input_path:
```

```
"/home/jonathanmoreirac/models/research/object_detection/training_no
vo/data/test.record"
  }
}
```