UNIVERSIDADE FEDERAL DE GOIÁS / INSTITUTO DE INFORMÁTICA

Modelos Fundacionais para Segmentação de Imagens

Uma Jornada do Clássico aos Modelos Fundacionais

Gustavo Rodrigues da Silva

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

UNIVERSIDADE FEDERAL DE GOIÁS (UFG)
INSTITUTO DE INFORMÁTICA (INF)

GUSTAVO RODRIGUES DA SILVA

Modelos Fundacionais para a Segmentação de Imagens

Uma Jornada do Clássico aos Modelos Fundacionais

Goiânia
2025

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TRABALHO DE CONCLUSÃO DE CURSO DE GRADUAÇÃO NO REPOSITÓRIO INSTITUCIONAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio do Repositório Institucional (RI/UFG), regulamentado pela Resolução CEPEC no 1240/2014, sem ressarcimento dos direitos autorais, de acordo com a Lei no 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo dos Trabalhos de Conclusão dos Cursos de Graduação disponibilizado no RI/UFG é de responsabilidade exclusiva dos autores. Ao encaminhar(em) o produto final, o(s) autor(a)(es)(as) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do Trabalho de Conclusão de Curso de Graduação (TCCG)

Nome(s) completo(s) do(a)(s) autor(a)(es)(as): GUSTAVO RODRIGUES DA SILVA

Título do trabalho: Modelos Fundacionais para a Segmentação de Imagens

Uma Jornada do Clássico aos Modelos Fundacionais

2. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador) Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante: a) consulta ao(à)(s) autor(a)(es)(as) e ao(à) orientador(a); b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo do TCCG. O documento não será disponibilizado durante o período de embargo.

Casos de embargo:

- Solicitação de registro de patente;

- Submissão de artigo em revista científica;

- Publicação como capítulo de livro.

Obs.: Este termo deve ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Gustavo Rodrigues Da Silva**, **Discente**, em 13/01/2025, às 22:29, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Fernando Marques Federson**, **Professor do Magistério Superior**, em 15/01/2025, às 16:16, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **5089778** e o código CRC **4F2B5D1D**.

Referência: Processo nº 23070.001587/2025-11 SEI nº 5089778

GUSTAVO RODRIGUES DA SILVA

Modelos Fundacionais para a Segmentação de Imagens

Uma Jornada do Clássico aos Modelos Fundacionais

Relatório final de Trabalho de Conclusão de Curso, apresentado à Universidade Federal de Goiás, como parte das exigências para a obtenção do título de Bacharel em Inteligência Artificial.
Orientador: Prof. Dr. Fernando Marques Federson

Goiânia
2025

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

SILVA, GUSTAVO RODRIGUES DA
Modelos Fundacionais para a Segmentação de Imagens [manuscrito] : Uma Jornada do Clássico aos Modelos Fundacionais / GUSTAVO RODRIGUES DA SILVA. - 2025.
93 f.

Orientador: Prof. Dr. Fernando Marques Federson.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Goiás, Instituto de Informática (INF), Inteligência Artificial, Goiânia, 2025.

1. inteligência artificial. 2. modelos fundacionais. 3. segmentação de imagens. I. Federson, Fernando Marques , orient. II. Título.

CDU 004

GUSTAVO RODRIGUES DA SILVA

Modelos Fundacionais para a Segmentação de Imagens
Uma Jornada do Clássico aos Modelos Fundacionais

Relatório final de Trabalho de Conclusão de Curso, apresentado à Universidade Federal de Goiás, como parte das exigências para a obtenção do título de Bacharel em Inteligência Artificial.

Data da Aprovação: 17 de dezembro de 2024.

Prof. Dr. Fernando Marques Federson
Orientador (INF-UFG)

Prof. Dr. Aldo André Díaz Salazar
Coordenador de TCC do BIA (INF-UFG)

Prof. Dr. Anderson da Silva Soares
Coordenador do BIA (INF-UFG)

Prof. Dr. Iwens Gervasio Sene Junior
(INF-UFG)

GUSTAVO RODRIGUES DA SILVA

Modelos Fundacionais para a Segmentação de Imagens

Uma Jornada do Clássico aos Modelos Fundacionais

RESUMO

Este Relatório de Conclusão de Curso tem como objetivo reunir os resultados da minha jornada para me tornar um especialista em **Visão Computacional (Segmentação de Imagens)**. Uma ilustração e sua narrativa descrevem os períodos de trabalho. Os Apêndices contêm os Termos de Aceite de Entrega e os resultados obtidos durante cada período de trabalho.

Palavras-chave: inteligência artificial, modelos grandes de linguagem, geração automática de datasets.

ABSTRACT

This Course Completion Report aims to bring together the results of my journey to become an expert in **Computer Vision (Image Segmentation)**. An illustration and its narrative describe the work periods. The Appendices contain the Delivery Acceptance Terms and the results obtained during each work period.

Keywords: artificial intelligence, large language models, automatic dataset generation.

Goiânia

2025

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Minha Jornada

Gustavo Rodrigues da Silva

Especialista em: Visão Computacional (Segmentação de Imagens)

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

MINHA JORNADA

Nome: Gustavo Rodrigues da Silva

Especialidade: Visão Computacional (Segmentação de Imagens)

Objetivo deste documento

Durante o processo da disciplina Residência em IA[1], foram gerados diversos resultados na construção da minha especialização. A cada semana, um conjunto de resultados foi formalizado por um Termo de Aceite de Entrega e avaliado por uma banca, considerando o planejado e o realizado para o período. Este documento tem como objetivo descrever esses resultados obtidos, fazendo referência aos Termos de Aceite de Entrega e seus documentos associados.

Minha Jornada

Minha jornada começou na **Semana 1** com a busca por artigos através do congresso CSCE 2024 e também no Google Scholar. Durante essa etapa inicial, concentrei-me na identificação de materiais voltados para a área da saúde, reconhecendo que a visão computacional tinha aplicações significativas neste domínio. Nesse processo, a ideia de trabalhar com "detecção de padrões em imagens" emergiu como um foco preliminar, representando um campo amplo e interconectado com diversas subáreas. Já na **Semana 2**, busquei delimitar minha área de atuação dentro da visão computacional, movendo-me além da aplicação prática e buscando uma compreensão mais técnica e específica. Durante essa busca, explorei eventos como CVPR, ECCV e ICCV, que me permitiram entender que "detecção de padrões em imagens" englobava praticamente todo o campo da visão computacional. Essa descoberta foi essencial para refinar meu foco, levando-me a optar pela segmentação de imagens como área de especialização. Com a clareza dessa decisão, iniciei uma busca por referências teóricas, incluindo livros e artigos que abordavam métodos

[1] Dez semanas, entre setembro de 2024 e dezembro de 2024.

clássicos e sua evolução histórica. Os materiais relacionados a estas duas Semanas podem ser encontrados no **Apêndice 1**.

Na **Semana 3**, meu objetivo foi aprofundar os estudos sobre segmentação de imagens, utilizando capítulos de livros que exploravam a história e os avanços dessa técnica ao longo do tempo. Essa leitura forneceu uma base sólida para entender como os métodos clássicos surgiram e evoluíram, além de destacar a importância da segmentação em contextos aplicados e acadêmicos. Apesar de valiosas, as referências literárias possuíam limitações temporais, pois não refletiam os avanços mais recentes da área. Com isso em mente, na **Semana 4**, voltei meu foco para pesquisas atuais, explorando o estado da arte em segmentação de imagens por meio de artigos e surveys que cobriam tanto métodos tradicionais quanto modernos. Um destaque foi um survey abrangente sobre modelos fundacionais, que introduziu conceitos-chave como treinamento em larga escala, auto-supervisão e propriedades emergentes. Essa etapa também marcou o início de experimentos mais avançados, onde testei algoritmos clássicos para comparar desempenho e preparar o terreno para avaliações futuras com modelos mais recentes. Os materiais relacionados a estas duas Semanas podem ser encontrados no **Apêndice 2**.

Na **Semana 5**, a descoberta de novos surveys aprofundou minha compreensão sobre os Modelos Fundacionais, que estavam se consolidando como o estado da arte em segmentação de imagens. Esses surveys destacaram características marcantes desses modelos, como a versatilidade, o uso de auto-supervisão, e a eliminação da necessidade de treinamentos específicos para cada tarefa. Essa etapa foi enriquecida pela exploração prática de exemplos no GitHub relacionados ao modelo SAM (Segment Anything Model), incluindo demonstrações interativas que destacavam sua capacidade de segmentação com prompts textuais e visuais. Com base nesses insights, planejei testar a viabilidade de aplicar esses modelos em diferentes domínios, ampliando meu entendimento sobre suas capacidades e limitações. Na **Semana 6**, a integração dos aprendizados teóricos e práticos tomou forma mais concreta. Explorei colabs fornecidos pela Meta para o modelo SAM, avaliando suas aplicações em imagens e vídeos. Paralelamente, decidi ampliar meu leque de ferramentas, documentando frameworks e modelos promissores como DINOv2, CLIP, e SigLIP, que foram escolhidos com base em suas potenciais contribuições para tarefas de

segmentação. Os materiais relacionados a estas duas Semanas podem ser encontrados no **Apêndice 3**.

Na **Semana 7**, iniciei um novo ciclo de experimentos, voltando minha atenção para modelos como DINOv2 e CLIPSeg. Realizei testes iniciais com essas ferramentas para segmentação de imagens, tentando segmentar diretamente, sem nenhum tipo de ajuste e utilizando os modelos individualmente. No entanto, os resultados desses testes não foram satisfatórios. Em seguida, decidi realizar um fine-tuning no DINOv2, mas também não obtive sucesso. Esse estágio foi marcado por algumas frustrações, o que implicou em mudanças de direcionamento na **Semana 8**. Durante essa Semana, meu foco foi expandir a integração entre modelos e avançar no fine-tuning do SAM, com o objetivo de estabelecer um comparativo robusto entre suas versões originais e ajustadas. Também explorei abordagens zero-shot com o modelo CLIPSeg, que se mostrou uma alternativa promissora para segmentações em domínios diversos, sem a necessidade de ajustes prévios. Além disso, investiguei possibilidades de combinar múltiplos modelos, conforme descrito em artigos e blogs que abordavam suas implementações colaborativas. Essa etapa reforçou a importância de testar diferentes configurações e metodologias para maximizar a eficiência e adaptabilidade dos modelos de segmentação, principalmente em tarefas que exigem alta precisão e flexibilidade em diferentes contextos. Os materiais relacionados a estas duas Semanas podem ser encontrados no **Apêndice 4**.

Na **Semana 9**, minha jornada tomou um rumo mais experimental, ao iniciar a integração entre dois dos modelos mais promissores que explorei até o momento: o SAM (Segment Anything Model) e o GroundingDINO (uma versão do DINO). Contudo, esse processo não trouxe resultados significativos para além do zero-shot, pois não consegui concluir sua implementação. Além disso, percebi a necessidade de realizar comparações mais sistemáticas entre os modelos que já havia testado ou estava testando. Com isso em mente, comecei uma abordagem comparativa, utilizando modelos como a U-Net com ResNet18, SAM e SAM2 em suas versões sem ajuste. Esses testes iniciais revelaram insights importantes sobre o desempenho relativo de cada modelo, ajudando a direcionar os próximos passos. Paralelamente, iniciei a elaboração de um documento analítico, "Análise Comparativa de Modelos de Segmentação", que registrava essas descobertas e fornecia

uma base sólida para futuros refinamentos. Na **Semana 10**, intensifiquei os testes, com foco no fine-tuning dos modelos SAM e SAM2. Durante essa etapa, adicionei um novo dataset mais complexo que atendia ao meu interesse em explorar a visão computacional no campo de aplicação da saúde. Concluí o documento de comparação entre os modelos escolhidos, alcançando resultados interessantes com o dataset complexo, composto por amostras de microscopia. Além disso, estabeleci os próximos passos da minha jornada, que continuará após a conclusão deste trabalho. Para isso, planejei direcionar meus esforços para a segmentação de vídeos, um campo que já havia explorado de forma preliminar, mas ainda não testado em larga escala. Os materiais relacionados a estas duas últimas Semanas podem ser encontrados no **Apêndice 5**.

Ao refletir sobre toda a experiência vivida nesta jornada, é evidente que os ganhos em termos de conhecimento e habilidades foram extraordinários. Trabalhei em uma área completamente nova para mim e consegui alcançar um desenvolvimento significativo tanto técnico quanto pessoal. Além disso, progredi em diversas outras competências: a prática de falar em público durante os Gates foi especialmente transformadora, ajudando-me a superar um ponto que eu considerava uma fraqueza. Também houve um avanço substancial no meu nível de inglês, impulsionado pela necessidade de realizar pesquisas aprofundadas sobre Modelos Fundacionais. Sinto que, em apenas 10 semanas, evoluí de uma forma que, em condições normais, provavelmente levaria anos para alcançar. Essa jornada não foi apenas uma oportunidade de aprendizado técnico, mas também um marco no meu crescimento profissional e pessoal.

APÊNDICE 1

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 19 de set. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Nessas primeiras semanas, o foco foi a identificação da minha área de especialização. O documento Stage01_Gustavo_Rodrigues detalha esse processo, o qual resultou na criação de duas pastas com conjuntos de artigos:

- Artigos_Residência_Etapa01 Contém artigos obtidos através da pesquisa nos artigos do CSCE.
- Artigos_Residência_Etapa02 Contém artigos obtidos por meio de uma pesquisa mais ampla, focada em visão computacional aplicada à saúde.

- Ao final desse processo, decidi que a minha área de especialização será **Detecção de Padrões em Imagens.**

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Para a próxima semana, planejo:

- Aprofundar o estudo sobre **Detecção de Padrões em Imagens**, com foco inicial na **história e nos fundamentos**.
- Realizar um levantamento bibliográfico que aborde a aplicação dessa técnica em **diferentes setores**.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

[Stage01_Gustavo_Rodrigues.doc citado no Termo de Aceite de Entrega de 19 de Setembro]

Nome: Gustavo Rodrigues da Silva

Matrícula: 202005481

Processo de Especialização em Análise de Padrões em Imagens

Meu processo de especialização começou com um enfoque amplo em **Inteligência Artificial (IA) aplicada à saúde**, abarcando diferentes subáreas e aplicações. Inicialmente, conduzi uma busca de artigos que exploram diversas vertentes da IA em saúde, resultando em **59 artigos**. Estes artigos discutiam desde o uso de IA para diagnósticos clínicos até a aplicação de machine learning em imagens médicas, cobrindo áreas como **detecção de câncer**, **análise de imagem por ultrassom**, e **diagnósticos assistidos por IA**.

Na **segunda etapa**, aprofundei meu estudo em **visão computacional**, com um foco especial em **análise de imagens**. Foram identificados **119 artigos**, organizados em subpastas de acordo com a área médica de aplicação. Durante essa fase, artigos sobre técnicas como **classificação de lesões de pele utilizando redes neurais convolucionais (CNNs)**, **segmentação de imagens de câncer** e **detecção de objetos em vídeos de ultrassom** se destacaram como contribuições centrais no campo da visão computacional.

Um exemplo notável foi o uso de **segmentação de imagens e reconhecimento de padrões** em diagnósticos dermatológicos, que foram abordados em artigos como o de detecção automática de **melanoma** usando deep learning e CNNs. Além disso, técnicas de **rastreamento de objetos** e **reconstrução 3D** foram aplicadas em estudos que envolvem a navegação e visualização de estruturas internas em exames de ultrassom fetal.

Conforme fui me aprofundando, ficou claro que a **Análise de Padrões em Imagens** seria o caminho ideal para me especializar, considerando que essa técnica não só é amplamente aplicável na saúde, mas também em áreas como **automação**, **segurança**, e **indústria**. A identificação e extração de padrões visuais em imagens são essenciais para resolver problemas complexos, como detecção precoce de doenças, segurança em ambientes industriais e até no reconhecimento de faces e objetos em sistemas de vigilância.

Portanto, minha especialização será voltada para a **Análise de Padrões em Imagens**, utilizando metodologias de **visão computacional** para lidar com a detecção de estruturas e

padrões visuais em diferentes contextos, com foco tanto em aplicações médicas quanto em setores variados.

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 26 de set. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

As atividades realizadas durante o Stage 02 foram:

- **Busca por material para identificar a área de atuação dentro de "Detecção de Padrões em imagens":**
  - CVPR (Conference on Computer Vision and Pattern Recognition)
    - Recognition: Categorization, detection, retrieval
    - Segmentation, grouping and shape analysis
  - ECCV (European Conference on Computer Vision)
    - Recognition and classification
    - Segmentation, grouping, and shape
  - ICCV (International Conference on Computer Vision)
    - Recognition: Categorization
    - Recognition: Detection
    - Recognition: Retrieval
    - Segmentation, grouping and shape analysis

- **Busca por materiais de apoio para a área específica de Segmentação:**
  - **Artigo:**
    - Image segmentation evaluation a survey of methods.pdf
  - **Livros:**
    - Pattern Classification, 2nd Edition.pdf
    - Digital Image Processing - Rafael C.Gonzalez 4th.pdf
    - Computer Vision_ Algorithms and Applications - Szeliski Primeira Edição.pdf
    - Computer Vision_ Algorithms and Applications - Szeliski Segunda Edição.pdf

- **Documento detalhando o Stage 02: Stage02_Gustavo_Rodrigues**

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- Continuar a leitura dos capítulos que são focados em Segmentação de imagens;
- Fazer alguns testes dos algoritmos clássicos de Segmentação de imagens;
- Entender como funciona a avaliação das Segmentações de imagens.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

[Stage02_Gustavo_Rodrigues citado no Termo de Aceite de Entrega de 26 de Setembro]

Introdução e Reflexão Inicial

A semana começou com uma reflexão sobre o que, de fato, constitui a **Detecção de Padrões em Imagens**. Minha pesquisa inicial me levou a explorar um livro voltado para detecção de padrões em geral *Pattern Classification, 2nd Edition.pdf*, que utiliza muitos exemplos visuais, mas ainda não me oferecia um caminho específico a seguir. Isso me fez perceber que deveria direcionar minha busca para algo mais específico dentro da área de **Visão Computacional**. Nesse ponto, lembrei-me que na primeira semana da residência, o professor Fernando Federson nos enviou alguns links para congressos. Esse insight me levou a pesquisar por congressos de IA que pudessem definir e aprofundar a noção de detecção de padrões em imagens.

Exploração em Congressos

Durante essa busca, encontrei três congressos renomados internacionalmente que abordam diretamente tópicos de visão computacional. Os três congressos identificados foram:

1. **CVPR (Conference on Computer Vision and Pattern Recognition)**
   - *Recognition: Categorization, Detection, Retrieval*
   - *Segmentation, Grouping, and Shape Analysis*
2. **ECCV (European Conference on Computer Vision)**
   - *Recognition and Classification*
   - *Segmentation, Grouping, and Shape*
3. **ICCV (International Conference on Computer Vision)**
   - *Recognition: Categorization, Detection, Retrieval*
   - *Segmentation, Grouping, and Shape Analysis*

Aproximação com Segmentação de Imagens

A partir dessas descobertas, comecei a explorar artigos e trabalhos que se alinhavam com as palavras-chave definidas nesses congressos. Nesse processo, deparei-me com muitas aplicações interessantes de segmentação, algumas bastante específicas e com forte impacto prático. Isso despertou ainda mais o meu interesse pela área de **Segmentação de Imagens**, pois vi um grande potencial na utilização dessas técnicas para solucionar problemas complexos.

Foi nesse momento que encontrei uma pesquisa bastante abrangente, Image segmentation evaluation a survey of methods.pdf, que consolidou meu interesse em me aprofundar ainda mais nas técnicas e métodos de segmentação. Uma observação

importante feita durante essa pesquisa foi que a **segmentação** não é um campo que depende exclusivamente de IA, ela possui raízes em fundamentos mais antigos de processamento de imagens.

Busca por Livros e Fundamentos Históricos

Com essa nova perspectiva, decidi explorar mais a fundo a história e os fundamentos da segmentação de imagens. A busca me levou a três livros que tratam não apenas da segmentação, mas também dos conceitos básicos de **Processamento de Imagens** e **Visão Computacional**.

- *Digital Image Processing - Rafael C.Gonzalez 4th.pdf*
- *Computer Vision_ Algorithms and Applications - Szeliski Primeira Edição.pdf*
- Computer Vision_ Algorithms and Applications - Szeliski Segunda Edição.pdf

Embora eu tenha iniciado a leitura dos capítulos específicos sobre segmentação, ainda há um vasto conteúdo a ser explorado, especialmente sobre as diferentes abordagens e técnicas aplicadas à área.

Próximos Passos

Com base na pesquisa realizada até agora, delineei os seguintes passos para as próximas semanas:

- Continuar a leitura dos capítulos focados em **Segmentação de Imagens**.
- Realizar testes com **algoritmos clássicos de Segmentação de Imagens**, para compreender as nuances práticas das diferentes abordagens.
- Explorar e entender como funcionam as **métricas de avaliação** nas segmentações de imagens.

APÊNDICE 2

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 2 de out. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o **Stage 03:**

Fiz a leitura de alguns capítulos dos livros voltados para a segmentação de imagens e resumi o conteúdo com o objetivo de entender como a segmentação teve início: Stage 03

Encontrei dois artigos de revisão que apresentam um panorama abrangente sobre a evolução da segmentação de imagens:
Image segmentation review Theoretical background and recent advances.pdf
Techniques and Challenges of Image Segmentation-A Review.pdf

Além disso, encontrei um artigo que aborda os métodos atualmente em desenvolvimento:
Image_Segmentation_in_Foundation_Model_Era_A_Survey.pdf

Realizei alguns testes com os métodos clássicos de segmentação: Stage03Testes.ipynb

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- Buscar mais artigos que abordem esses métodos novos.
- Analisar, entender e resumir sobre as ferramentas usadas nos trabalhos pesquisados.
- Realizar testes com métodos mais avançados de segmentação.

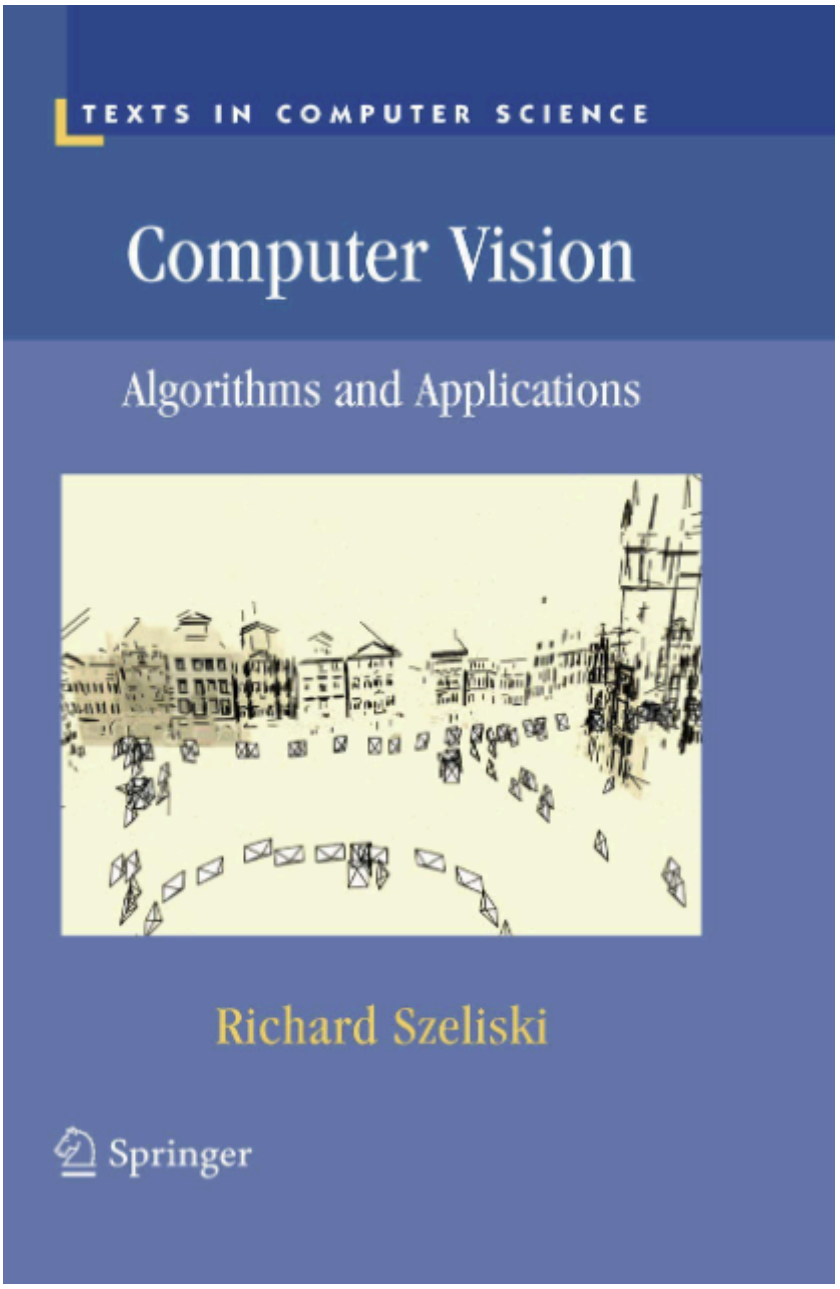
Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

[Stage03.doc citado no Termo de Aceite de Entrega de 02 de Outubro]



Resumo Capítulo 5: Segmentação

A segmentação de imagem é um processo essencial em visão computacional, responsável por dividir uma imagem em partes distintas, facilitando a análise de objetos e suas características. No capítulo 5 do livro *Computer Vision: Algorithms and Applications*, Richard

Szeliski explora várias técnicas de segmentação, cada uma adequada a diferentes tipos de imagens e cenários.

1. Contornos Ativos (Active Contours)

Esta técnica envolve ajustar uma curva (conhecida como "snake") para capturar os contornos dos objetos dentro de uma imagem. A ideia principal é que essa curva se mova para minimizar uma função de energia composta por forças internas (que mantêm a curva suave) e forças externas (que atraem a curva para os limites dos objetos, como gradientes de imagem).

- **Snakes**: A abordagem clássica dos contornos ativos utiliza gradientes de imagem para mover a curva em direção às bordas de um objeto, minimizando uma função de energia. Esse método é útil para detecção de bordas em imagens estáticas.
- **Snakes Dinâmicos e CONDENSATION**: Essa extensão dos snakes clássicos incorpora um modelo probabilístico para rastrear objetos em movimento ao longo de sequências de imagens, como vídeos. O método de propagação de densidade condicional (CONDENSATION) utiliza partículas para prever e rastrear a evolução de contornos, lidando bem com ruídos e mudanças de forma.
- **Scissors Interativos**: Esta técnica combina segmentação automática com intervenção manual, onde o usuário pode guiar o processo de delineamento de contornos. É particularmente útil em imagens onde a segmentação totalmente automatizada não é confiável.
- **Level Sets**: Os level sets são uma técnica avançada para modelar mudanças complexas na topologia dos objetos (como fusões e divisões). Essa abordagem é baseada em uma função implícita que evolui ao longo do tempo, permitindo segmentar objetos que mudam de forma ou que se fundem e se dividem.

Aplicação: As técnicas de contorno ativo são amplamente usadas em rastreamento de objetos em vídeos e rotoscopia em animações.

2. Divisão e Fusão de Regiões (Split and Merge)

Esse método particiona uma imagem em regiões menores com base em características estatísticas locais. A ideia central é que uma imagem pode ser dividida (split) em regiões até que os critérios de homogeneidade sejam atendidos, ou fundida (merge) agrupando regiões adjacentes com propriedades semelhantes.

- **Watershed**: Inspirada pela geografia, a técnica de watershed trata a segmentação como uma simulação de alagamento, onde os "picos" de intensidade da imagem são vistos como bacias hidrográficas. Esse método é amplamente utilizado em imagens médicas e de microscopia.

- **Clustering Divisivo e Aglomerativo**: Essas duas abordagens envolvem dividir (divisivo) ou fundir (aglomerativo) regiões com base em propriedades de similaridade. A segmentação gráfica, por exemplo, é uma técnica em que as regiões são divididas ou agrupadas em diferentes níveis hierárquicos.

3. Mean Shift e K-means

Ambas as técnicas são métodos de clustering, que agrupam pixels com base em suas características. Enquanto o K-means agrupa os pixels em torno de centróides, o Mean Shift é mais flexível, encontrando modos em distribuições de densidade sem a necessidade de especificar o número de clusters.

- **K-means**: Um algoritmo de clustering amplamente utilizado para segmentar imagens em que os pixels são agrupados em torno de centróides definidos. É uma das formas mais simples de segmentação baseada em clusters, mas sua simplicidade vem com a desvantagem de que o número de clusters precisa ser definido antecipadamente.
- **Mean Shift**: Um algoritmo não paramétrico que encontra os modos em uma distribuição de densidade, deslocando um kernel através da imagem até encontrar regiões de maior densidade. Isso faz com que o Mean Shift seja flexível e eficiente para lidar com imagens com grupos não lineares de pixels.

4. Cortes Normalizados (Normalized Cuts)

Proposto por Shi e Malik (2000), o algoritmo de cortes normalizados é baseado em grafos e usa uma função de corte para particionar uma imagem em regiões com alta afinidade interna (similaridade entre os pixels) e baixa afinidade externa (diferença entre grupos de pixels). Esta técnica é amplamente utilizada, mas pode ser computacionalmente intensiva devido à necessidade de resolver problemas de autovalores.

Exemplo: Esta técnica é eficaz para segmentar grandes áreas homogêneas em imagens complexas, mas requer bastante processamento devido ao cálculo de matrizes de afinidade.

5. Cortes em Grafos e Métodos Baseados em Energia

Os métodos baseados em energia envolvem a minimização de uma função de custo que reflete a suavidade da segmentação e a consistência interna das regiões. Os cortes em grafos são uma abordagem combinatória para segmentar uma imagem, minimizando a função de energia, sendo especialmente útil em problemas de segmentação binária.

- **Graph Cuts**: Essa técnica trata a segmentação como um problema de corte de grafo, onde a imagem é representada como um grafo e o corte mínimo entre nós do

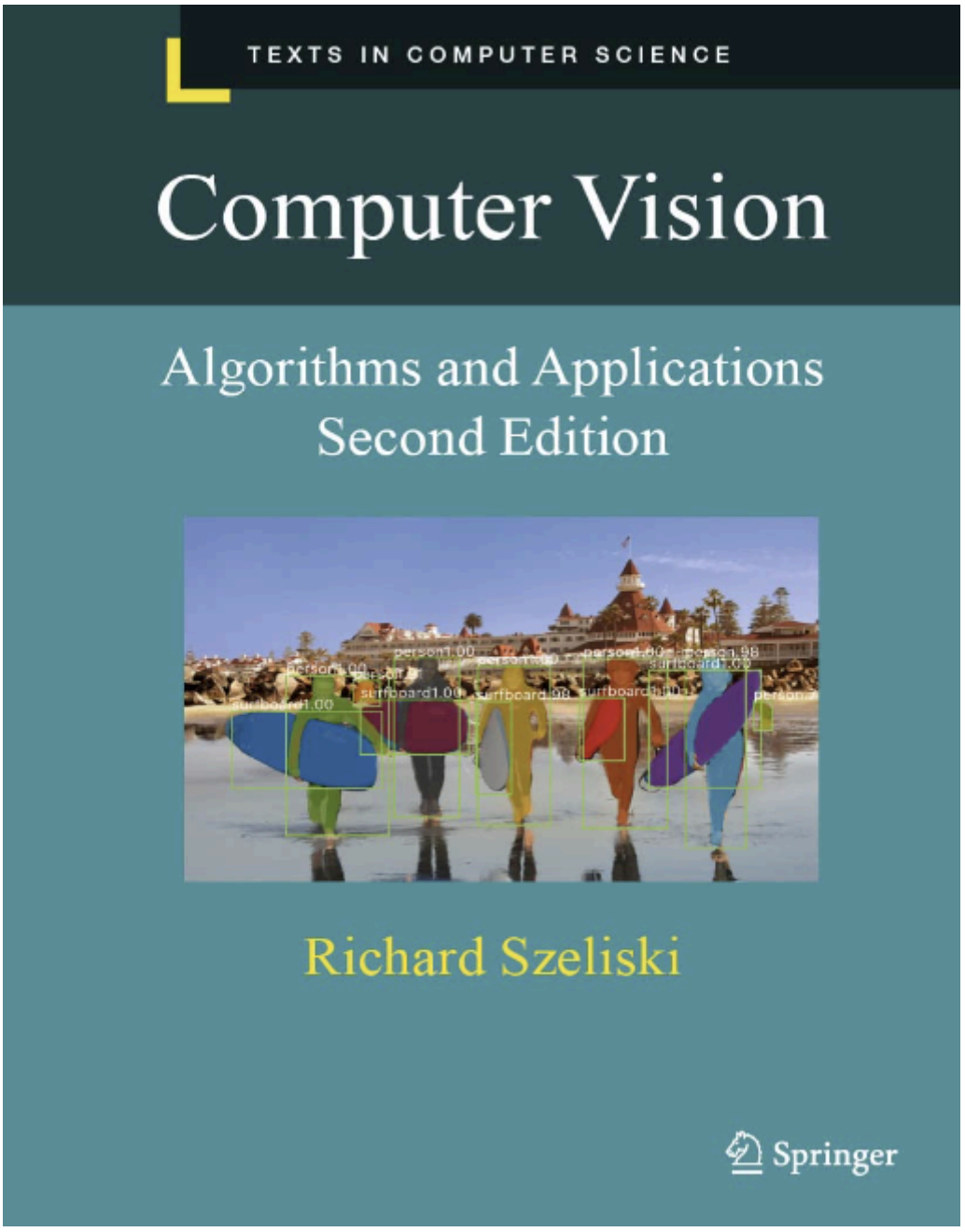
grafo resulta em uma segmentação ideal. É amplamente utilizado em segmentação binária.

- **GrabCut**: Uma extensão interativa do Graph Cuts, onde o usuário fornece uma estimativa inicial das regiões de fundo e primeiro plano. A segmentação é refinada automaticamente com base nos cortes de grafo, permitindo uma interação mais intuitiva e refinada com o usuário.

6. Aplicações em Segmentação Médica

As técnicas de segmentação são extremamente importantes em imagens médicas, como tomografias computadorizadas (CT) e ressonâncias magnéticas (MRI), onde é necessário isolar órgãos ou estruturas anatômicas. Métodos baseados em cortes de grafo são particularmente úteis aqui, pois permitem uma segmentação robusta em cenários complexos e com ruído.

Exemplo: Na segmentação de imagens médicas, essas técnicas podem ser combinadas com aprendizado profundo, proporcionando resultados mais precisos e automatizados.



Resumo Capítulo 06 Recognition

6.4 Segmentação Semântica

A segmentação semântica é uma forma avançada de reconhecimento de objetos e compreensão de cena, que envolve a classificação de cada pixel de uma imagem com base

em sua classe. Esse processo resulta em uma representação por pixel do conteúdo da imagem.

- **Abordagens iniciais**: Historicamente, abordagens como a minimização de energia e inferência bayesiana (como campos aleatórios condicionais, CRFs) eram comuns. Sistemas como o *TextonBoost* usavam potenciais unários e pares para segmentação, com classificadores de textura e layout.
- **Redes Convolucionais Completas (FCNs)**: A introdução de redes convolucionais completas (FCNs) permitiu a rotulagem por pixel usando redes neurais, melhorando a precisão na segmentação semântica. Essas redes foram aprimoradas com técnicas como CRFs condicionais, upsampling deconvolucional e conexões finas em redes U-Net, todas voltadas para melhorar a resolução e a precisão da segmentação.
- **Arquiteturas modernas**: A maioria dos sistemas de segmentação atuais é construída com base em arquiteturas como *Feature Pyramid Networks* (FPN), que usa conexões de cima para baixo para integrar informações semânticas em mapas de alta resolução. Redes como a *Pyramid Scene Parsing Network* (PSPNet) utilizam *pooling* piramidal para agregar características em vários níveis de resolução.

6.4.1 Aplicação: Segmentação de Imagens Médicas

A segmentação de imagens médicas é uma das aplicações mais promissoras da segmentação semântica, sendo utilizada para identificar e isolar tecidos anatômicos para análise quantitativa posterior. Inicialmente, técnicas de otimização como campos aleatórios de Markov e classificadores discriminativos (como florestas aleatórias) eram utilizadas. Recentemente, a área migrou para abordagens de aprendizado profundo, que demonstraram melhores resultados na segmentação de tumores cerebrais, entre outras aplicações.

6.4.2 Segmentação de instâncias

A segmentação de instâncias é a tarefa de encontrar todos os objetos relevantes em uma imagem e gerar máscaras pixel-precisas para suas regiões visíveis. Métodos iniciais dependiam do reconhecimento de instâncias de objetos conhecidos e projeção de modelos 3D rígidos na cena. No entanto, com objetos flexíveis (como humanos), métodos baseados em correspondências de características foram desenvolvidos.

- **Máscara R-CNN**: Uma grande evolução ocorreu com a introdução da *Mask R-CNN*, que adiciona uma ramificação adicional de segmentação ao já existente *Faster R-CNN*, permitindo a segmentação precisa de objetos, além de refinamentos na caixa delimitadora e classificação. Esta técnica tornou-se um padrão em

segmentação de instâncias, sendo continuamente aprimorada por avanços em arquiteturas de backbone.

6.4.3 Segmentação panóptica

A segmentação panóptica combina segmentação semântica e de instâncias em uma única abordagem, permitindo rotular todos os objetos e "stuff" (materiais ou fundos) em uma cena. Este método visa fornecer uma segmentação completa de cada elemento da imagem, com uma métrica de qualidade panóptica (PQ) que equilibra a precisão tanto na segmentação de instâncias quanto na semântica. Redes como a *Panoptic Feature Pyramid Network* (PFPN) foram desenvolvidas para essa tarefa, combinando ramificações para segmentação de instâncias e semântica.

6.4.4 Aplicação: Edição de Fotos Inteligente

Os avanços no reconhecimento de objetos e na compreensão de cenas têm impulsionado ferramentas de edição de fotos inteligentes, permitindo uma edição semiautomática. Um exemplo é o sistema *Photo Clip Art*, que reconhece e segmenta objetos, permitindo que os usuários os adicionem às suas próprias fotos. Outro exemplo é o sistema de preenchimento de cenas, que utiliza vastos bancos de imagens para substituir pixels ausentes ou áreas removidas de uma imagem de maneira natural, por meio de técnicas como *Poisson image blending*.

6.4.5 Estimativa de pose

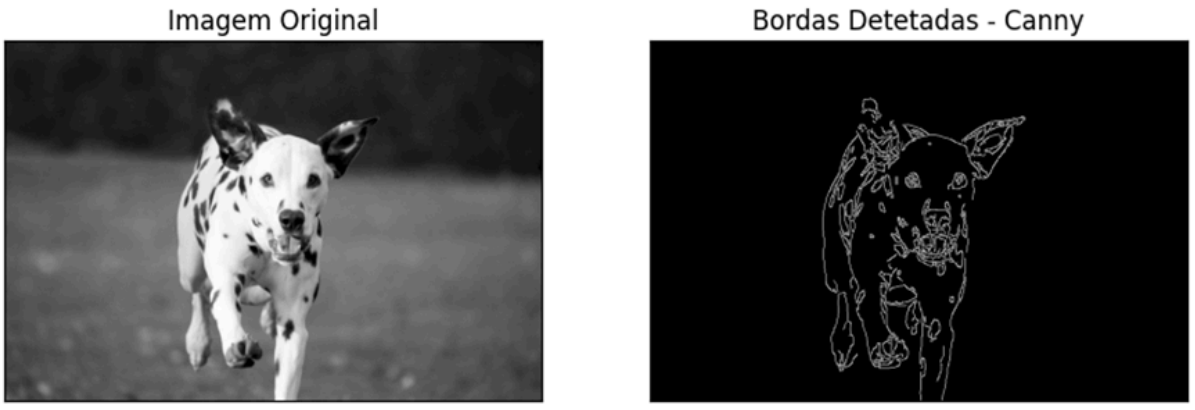
A estimativa de pose humana envolve a segmentação e identificação de pontos-chave do corpo, como a cabeça, tronco e membros. Iniciada com o trabalho de Felzenszwalb e Huttenlocher, a detecção e estimativa de pose evoluíram rapidamente com o advento das redes profundas. Hoje, a estimativa de pose é amplamente utilizada em várias aplicações de visão computacional, especialmente na detecção de poses em tempo real.

[Stage03Testes.ipynb citado no Termo de Aceite de Entrega de 02 de Outubro]

#Testes Métodos Clássicos

Detecção de Bordas - Canny

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Carregar a imagem
img = cv2.imread('/content/cachorro-preto-e-branco-1.png', 0)

# Aplicar o filtro Canny
edges = cv2.Canny(img, 100, 200)

# Exibir a imagem original e a borda detectada
plt.figure(figsize=(10,5))
plt.subplot(121), plt.imshow(img, cmap='gray')
plt.title('Imagem Original'), plt.xticks([]), plt.yticks([])
plt.subplot(122), plt.imshow(edges, cmap='gray')
plt.title('Bordas Detetadas - Canny'), plt.xticks([]), plt.yticks([])

plt.show()
```



Segmentação por Watershed

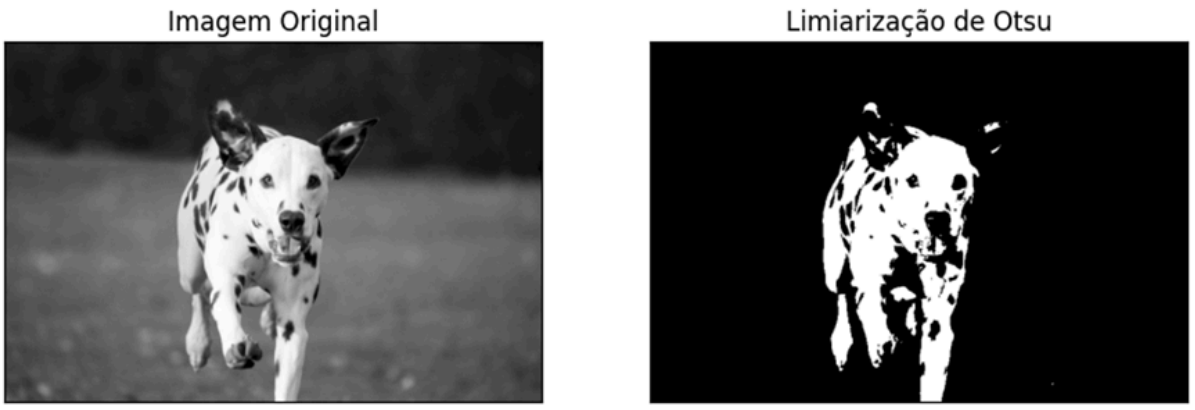
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

```
# Carregar a imagem
img = cv2.imread('/content/cachorro-preto-e-branco-1.png')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Aplicar limiarização
ret, thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY_INV +
cv2.THRESH_OTSU)

# Remover ruído
kernel = np.ones((3,3), np.uint8)
opening = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel, iterations=2)

# Determinar o fundo
sure_bg = cv2.dilate(opening, kernel, iterations=3)

# Determinar o primeiro plano
dist_transform = cv2.distanceTransform(opening, cv2.DIST_L2, 5)
ret, sure_fg = cv2.threshold(dist_transform, 0.7*dist_transform.max(),
255, 0)

# Encontrar a região desconhecida
sure_fg = np.uint8(sure_fg)
unknown = cv2.subtract(sure_bg, sure_fg)

# Aplicar Watershed
ret, markers = cv2.connectedComponents(sure_fg)
markers = markers + 1
markers[unknown == 255] = 0

markers = cv2.watershed(img, markers)
img[markers == -1] = [255, 0, 0]

# Exibir o resultado
plt.figure(figsize=(10,5))
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title('Segmentação por Watershed'), plt.xticks([]), plt.yticks([])

plt.show()
```

Limiarização Global - Otsu

```
import cv2
import matplotlib.pyplot as plt

# Carregar a imagem
img = cv2.imread('/content/cachorro-preto-e-branco-1.png', 0)

# Aplicar limiarização de Otsu
ret, thresh = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY +
cv2.THRESH_OTSU)

# Exibir a imagem original e a imagem binária
plt.figure(figsize=(10,5))
plt.subplot(121), plt.imshow(img, cmap='gray')
plt.title('Imagem Original'), plt.xticks([]), plt.yticks([])
plt.subplot(122), plt.imshow(thresh, cmap='gray')
plt.title('Limiarização de Otsu'), plt.xticks([]), plt.yticks([])
```

```
plt.show()
```



Segmentação por Crescimento de Regiões

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Carregar a imagem
img = cv2.imread('/content/cachorro-preto-e-branco-1.png', 0)

# Função de crescimento de regiões
def region_growing(img, seed):
      h, w = img.shape
      mask = np.zeros((h, w), np.uint8)
      mask[seed[1], seed[0]] = 255
      growing = True

      while growing:
      growing = False
      for x in range(1, h-1):
            for y in range(1, w-1):
                  if mask[x, y] == 255:
                     for dx in [-1, 0, 1]:
                         for dy in [-1, 0, 1]:
                              if mask[x + dx, y + dy] == 0 and
abs(int(img[x, y]) - int(img[x + dx, y + dy])) < 20:
                                    mask[x + dx, y + dy] = 255
```

```
                                growing = True
        return mask

    # Coordenadas da semente
    seed = (100, 100)

    # Aplicar o crescimento de regiões
    segmented_img = region_growing(img, seed)

    # Exibir a imagem original e a segmentada
    plt.figure(figsize=(10,5))
    plt.subplot(121), plt.imshow(img, cmap='gray')
    plt.title('Imagem Original'), plt.xticks([]), plt.yticks([])
    plt.subplot(122), plt.imshow(segmented_img, cmap='gray')
    plt.title('Segmentação por Crescimento de Regiões'), plt.xticks([]),
plt.yticks([])

    plt.show()
```

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 10 de out. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o **Stage 04:**

Coloquei as definições que dos métodos de segmentação que tinha lido em um documento para simplificar a busca: Resumo_Métodos de segmentação .

Iniciei testes com modelos baseados em redes neurais para explorar a diferença significativa entre esses métodos e os clássicos. Deixei dois testes disponíveis aqui: Stage04Testes.ipynb .

Ao longo desse processo, percebi que, nesses testes eu já estava levantando alguns frameworks e decidi documentá-los também: FrameworksStage04 .

Dando continuidade aos estudos sobre Modelos Fundacionais através do Survey: Image_Segmentation_in_Foundation_Model_Era_A_Surve.pdf .

Encontrei Segment Anything.pdf e o SAM 2 Segment Anything in Images and Videos.pdf o que me levou ao github, também descobri uma demo que demonstra a grande proficiência do modelo segmentado imagens e vídeo, o que direcionou meu estudo para essa área da segmentação de imagens.

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- Dar continuidade aos estudos sobre Modelos Fundacionais.
- Explorar frameworks adicionais além dos já documentados, com foco nos Modelos Fundacionais.
- Avaliar a viabilidade de aplicação desses Modelos Fundacionais.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

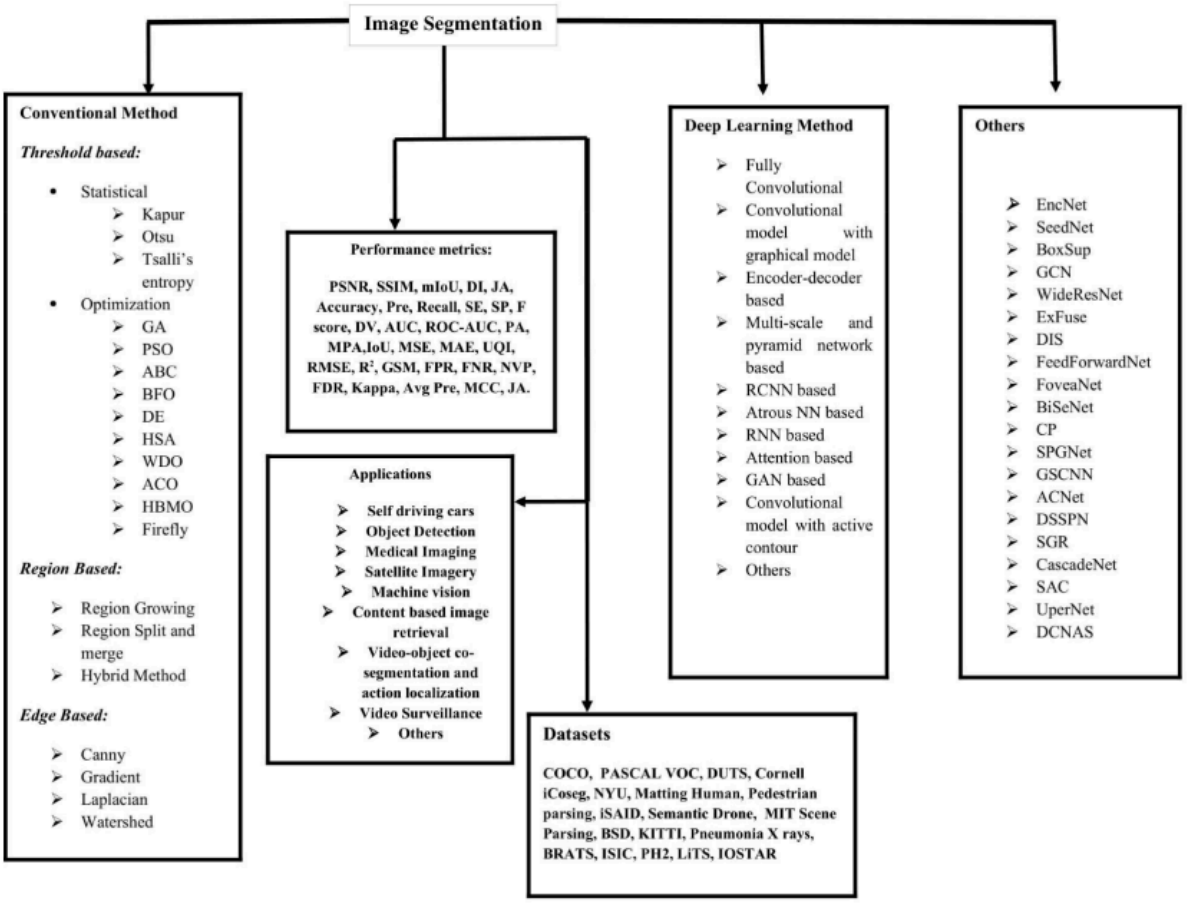
Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

[Resumo_Métodos de segmentação.doc citado no Termo de Aceite de Entrega de 10 de Outubro]

Resumo Métodos de segmentação

Definições baseadas no Survey:

Image segmentation review Theoretical background and recent adv…



Métodos Convencionais

Threshold-based Methods:
Métodos baseados em threshold (limiar) são uma das abordagens mais simples e amplamente utilizadas na segmentação de imagens. O princípio básico é selecionar

um valor de limiar para dividir os pixels de uma imagem em dois grupos: os pixels que pertencem ao objeto de interesse e os que pertencem ao fundo. Essa técnica é eficaz em imagens com contrastes claros entre o objeto e o fundo, como em imagens com histogramas bimodais (com dois picos claros).

- **Kapur**: O método de Kapur é uma abordagem baseada na maximização da entropia, onde o limiar ótimo é escolhido para maximizar a entropia da informação dentro das classes de pixels. Ele é particularmente útil para imagens com múltiplos níveis de cinza, pois maximiza a separação entre as informações do objeto e do fundo.

- **Otsu**: O método de Otsu é uma das técnicas mais conhecidas de segmentação por threshold. Ele seleciona automaticamente o limiar que minimiza a variância intra-classe, ou seja, o método busca reduzir a diferença dentro das classes, melhorando a separação entre o objeto e o fundo. Isso o torna adequado para imagens com histogramas bimodais, onde a separação entre objeto e fundo é bem definida.

- **Tsallis's entropy**: Esse método usa a entropia não-extensiva de Tsallis, uma generalização da entropia de Shannon, para lidar com imagens mais complexas. Ele é particularmente eficaz em cenários onde as imagens têm ruídos ou variações complexas de textura, e permite uma maior flexibilidade no ajuste do limiar com base nas características da imagem.

Region-based Methods:
Métodos baseados em regiões focam na segmentação de uma imagem dividindo-a em áreas contíguas com propriedades similares, como cor, textura ou intensidade. Esses métodos são úteis em casos onde os objetos de interesse têm características homogêneas que podem ser distinguidas de outras regiões.

- **Region Growing**: O crescimento de regiões começa com um ou mais pixels "semente" e expande essas regiões agregando novos pixels adjacentes que atendem a um critério de similaridade, como intensidade ou cor. Essa técnica

funciona bem quando o objeto de interesse possui características homogêneas e um bom contraste em relação ao fundo. No entanto, pode ser sensível ao ruído e exige uma boa escolha das sementes.

- **Region Split and Merge**: Esse método alterna entre dividir e fundir regiões da imagem. A divisão ocorre quando uma região é considerada heterogênea, e a fusão ocorre quando duas regiões adjacentes compartilham características semelhantes. Esse método combina a robustez de dividir e fundir regiões para lidar com variações complexas de características.

- **Hybrid Methods**: Métodos híbridos combinam abordagens de crescimento de regiões e fusão para superar as limitações individuais de cada técnica. Esses métodos podem lidar melhor com imagens complexas e são mais adaptáveis a diferentes tipos de imagem.

Edge-based Methods:
Métodos baseados em bordas (edges) detectam transições abruptas de intensidade na imagem, que frequentemente correspondem aos contornos dos objetos. A ideia é que as bordas formam a delimitação natural entre os objetos e o fundo.

- **Canny**: O algoritmo de Canny é amplamente utilizado na detecção de bordas. Ele é composto por várias etapas, incluindo a suavização da imagem com um filtro gaussiano para remover o ruído, a detecção do gradiente da imagem, e a aplicação de um processo de supressão de não-máximos para refinar a detecção de bordas. Por fim, utiliza limiares duplos para ligar bordas conectadas, resultando em uma segmentação de bordas precisa e contínua.

- **Gradient**: Os métodos de gradiente detectam bordas baseando-se nas mudanças abruptas de intensidade entre pixels adjacentes. Esses métodos geralmente usam operadores de gradiente, como Sobel ou Prewitt, para calcular a magnitude e a direção das bordas. Embora simples, eles podem ser sensíveis ao ruído e, por isso, muitas vezes são usados em combinação com técnicas de suavização.

- **Laplacian**: O método Laplaciano é uma técnica baseada na segunda derivada da intensidade da imagem, que identifica as mudanças rápidas de intensidade. Ele é mais sensível a variações de bordas finas, mas também pode amplificar o ruído. Frequentemente, o operador Laplaciano é aplicado após um processo de suavização para reduzir a sensibilidade ao ruído.

- **Watershed**: A técnica de watershed é um método inspirado em processos geográficos, onde a imagem é tratada como uma topografia, com bacias e cristas. O algoritmo identifica bacias e preenche as regiões com base em linhas de água imaginárias. É particularmente eficaz para imagens com objetos que têm contornos bem definidos, mas pode precisar de técnicas de pré-processamento, como filtragem de ruído ou marcação de sementes, para evitar supersegmentação.

Métodos de Aprendizado Profundo (Deep Learning Methods)

Os métodos de aprendizado profundo revolucionaram a segmentação de imagens, oferecendo uma precisão muito maior do que as abordagens convencionais. Em vez de depender de técnicas baseadas em características manuais, os modelos de deep learning aprendem automaticamente as melhores representações dos dados durante o treinamento. A principal vantagem desses métodos é sua capacidade de lidar com grandes quantidades de dados e descobrir padrões complexos de maneira eficiente.

- **Fully Convolutional Networks (FCN)**:
  As redes convolucionais totalmente convolucionais (FCN) são um dos primeiros métodos de deep learning aplicados à segmentação de imagens. Diferentemente das CNNs tradicionais, que fornecem um único valor de saída (por exemplo, uma classe para a imagem inteira), as FCNs produzem mapas de predição pixel-a-pixel. Isso é feito substituindo as camadas totalmente conectadas (presentes nas CNNs convencionais) por camadas convolucionais, permitindo que a rede mantenha a resolução espacial da imagem de entrada. O resultado é uma segmentação mais precisa e detalhada, onde cada pixel é classificado.

- **Convolutional Model with Graphical Model**:
  Este método combina redes convolucionais com modelos gráficos probabilísticos, como Campos Aleatórios de Markov (MRF) ou Campos Aleatórios Condicionais (CRF), para melhorar a segmentação. As CNNs são usadas para capturar

características locais da imagem, enquanto os modelos gráficos são usados para impor dependências espaciais entre os pixels, resultando em segmentações mais suaves e coerentes. Essa abordagem é útil para refinar bordas de objetos e garantir que regiões homogêneas sejam bem segmentadas.

- **Encoder-Decoder Based**:
  Os modelos baseados em arquiteturas de codificador-decodificador (como a U-Net) são amplamente utilizados para tarefas de segmentação semântica. O codificador reduz progressivamente a resolução da imagem, extraindo características de nível superior, enquanto o decodificador reconstrói a imagem com as características segmentadas. Esse processo é auxiliado por conexões de "skip" que transferem informações de alta resolução do codificador para o decodificador, permitindo que detalhes finos sejam preservados. Esse tipo de modelo é especialmente eficaz para tarefas médicas e imagens com muitos detalhes.

- **RCNN-based**:
  As Redes Convolucionais Baseadas em Regiões (Region-based Convolutional Neural Networks - RCNN) são projetadas para tarefas de detecção e segmentação de objetos. Elas propõem regiões candidatas na imagem e aplicam CNNs nessas regiões para identificar e segmentar objetos. O RCNN e suas variantes (como Fast RCNN e Mask RCNN) são amplamente utilizados para tarefas de segmentação instância, onde não apenas os objetos são classificados, mas também seus contornos são delineados de maneira precisa.

- **Atrous Network-based**:
  Redes baseadas em convoluções dilatadas (também chamadas de convoluções atrous) são projetadas para lidar com imagens de alta resolução sem reduzir o campo de visão. A dilatação dos filtros convolucionais permite que eles cubram uma área maior da imagem, capturando informações contextuais amplas sem aumentar o número de parâmetros. Isso torna essa abordagem ideal para segmentação de imagens com objetos em diferentes escalas, como em imagens médicas ou de satélite.

- **RNN-based**:
  Redes Neurais Recorrentes (RNNs) são comumente usadas para lidar com dados sequenciais, como vídeos, em tarefas de segmentação temporal. Elas podem capturar dependências temporais entre quadros consecutivos de um vídeo, permitindo segmentações mais consistentes ao longo do tempo. A Long Short-Term Memory (LSTM), uma variação de RNN, é frequentemente usada para capturar essas dependências de longo prazo.

- **Attention-based**:
  Modelos baseados em mecanismos de atenção são cada vez mais usados em segmentação de imagens. O mecanismo de atenção permite que a rede foque em partes específicas da imagem que são mais relevantes para a tarefa de segmentação, ignorando informações irrelevantes. Isso melhora a precisão ao capturar os detalhes mais importantes de objetos complexos. Modelos como o Transformer aplicam mecanismos de auto-atenção para aprender dependências globais e locais, o que é particularmente eficaz em segmentação de imagens com objetos sobrepostos.

- **GAN-based**:
  As Redes Gerativas Adversariais (GANs) têm sido aplicadas à segmentação como uma abordagem de aprendizado não supervisionado. Nesse caso, uma rede geradora cria segmentações de imagens e uma rede discriminadora tenta diferenciar as segmentações geradas das reais. O processo adversarial entre as duas redes leva a segmentações mais realistas e precisas. GANs são particularmente úteis em cenários onde os dados rotulados são escassos.

- **Convolutional Model with Active Contour**:
  Modelos convolucionais com contornos ativos integram técnicas clássicas de contorno ativo (snakes) em arquiteturas convolucionais. Isso permite que o modelo aprenda não apenas as características da imagem, mas também como ajustar os contornos para delinear os objetos de maneira precisa. Essa abordagem é útil em imagens onde os contornos são difíceis de detectar apenas com convoluções tradicionais.

[FrameworksStage04.doc citado no Termo de Aceite de Entrega de 10 de Outubro]

FrameworksStage04

1. PyTorch (Torchvision)

O PyTorch é uma biblioteca open-source de aprendizado profundo desenvolvida pelo Facebook AI Research. É amplamente utilizada por sua flexibilidade e facilidade de uso, especialmente para tarefas envolvendo processamento de imagens, visão computacional e processamento de linguagem natural. **Torchvision** é um pacote complementar ao PyTorch que fornece datasets, modelos pré-treinados e transformações para a visão computacional.

Documentação: Torchvision

2. OpenCV

OpenCV é uma biblioteca open-source usada para processamento de imagens e vídeos em tempo real. Ela é amplamente utilizada em projetos de visão computacional, desde operações básicas como manipulação de pixels até tarefas avançadas como detecção de objetos e rastreamento.

Documentação: OpenCV

3. Scikit-Image

Scikit-Image é uma coleção de algoritmos para processamento de imagens em Python, construída sobre as bibliotecas SciPy e NumPy. É especialmente útil para tarefas de segmentação, transformação, análise e manipulação de imagens, e é amplamente usada em pesquisa e desenvolvimento.

Documentação: Scikit-Image

4. Detectron2

Detectron2 é um framework de aprendizado profundo criado pelo Facebook AI Research para facilitar a implementação de modelos de detecção e segmentação de objetos. Ele fornece uma coleção de modelos de última geração, como Faster R-CNN, Mask R-CNN, e Panoptic Segmentation.

Documentação: Detectron2

[Stage04Testes.ipynb citado no Termo de Aceite de Entrega de 10 de Outubro]

Imports e Instalações

```
%%capture

# Instalações necessárias para o Detectron2
!pip install -U torch torchvision torchaudio
!pip install -U pycocotools
!pip install -U opencv-python

# Instala Detectron2 (versão compatível com a GPU do Colab)
!pip install git+https://github.com/facebookresearch/detectron2.git
import torch
import cv2
import os
import numpy as np
import matplotlib.pyplot as plt
from google.colab.patches import cv2_imshow
from detectron2.engine import DefaultPredictor
from detectron2.config import get_cfg
from detectron2 import model_zoo
from detectron2.utils.visualizer import Visualizer
from detectron2.data import MetadataCatalog
from detectron2.data import DatasetCatalog
```

Dados

```
%%capture
```

```
 !!gdown "1hF0euWiiaSZxq517InKyJXmngbgnqK97"
%%capture

 !!unzip "/content/images.zip"
```

Testes

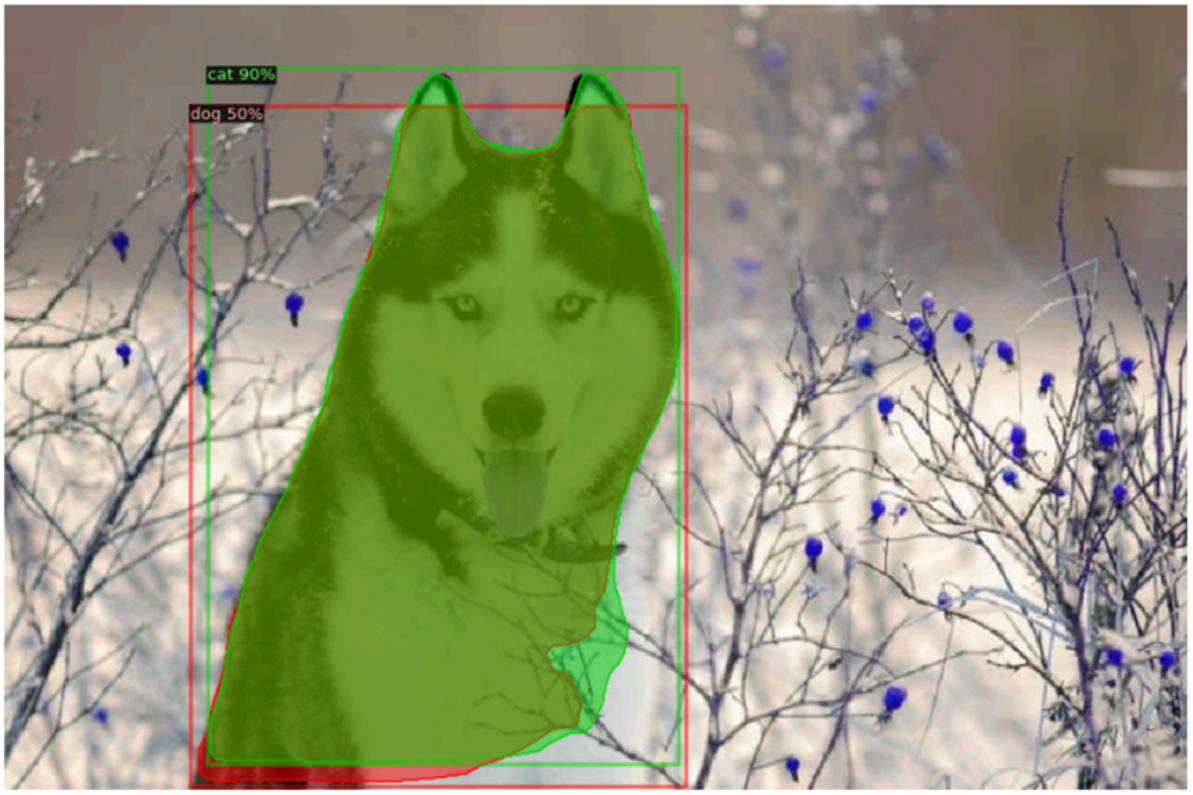
Mask R-CNN

```
 # Configuração do modelo Mask R-CNN usando Detectron2
 def setup_mask_rcnn():
      cfg = get_cfg()

cfg.merge_from_file(model_zoo.get_config_file("COCO-InstanceSegmentation/
mask_rcnn_R_50_FPN_3x.yaml"))
     cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.5  # Definir o threshold
     cfg.MODEL.WEIGHTS =
model_zoo.get_checkpoint_url("COCO-InstanceSegmentation/mask_rcnn_R_50_FP
N_3x.yaml")
     cfg.MODEL.DEVICE = "cuda" if torch.cuda.is_available() else "cpu"
      return cfg, DefaultPredictor(cfg)

 # Inicializar o preditor do Mask R-CNN e a configuração
 cfg, predictor = setup_mask_rcnn()

 image_folder = '/content/TestesSeg'

 # Percorrer todas as imagens na pasta
 for image_name in os.listdir(image_folder):
      if image_name.endswith(".jpg") or image_name.endswith(".png") or
image_name.endswith(".jpeg"):
      image_path = os.path.join(image_folder, image_name)

      # Carregar a imagem
      image = cv2.imread(image_path)

      # Aplicar o Mask R-CNN na imagem
      outputs = predictor(image)
```

```
      # Visualizar os resultados
      v = Visualizer(image[:, :, ::-1],
MetadataCatalog.get(cfg.DATASETS.TRAIN[0]), scale=1.2)
      v = v.draw_instance_predictions(outputs["instances"].to("cpu"))

      # Exibir a imagem segmentada com as máscaras
         plt.figure(figsize=(10, 10))
         plt.imshow(v.get_image()[:, :, ::-1])
      plt.title(f'Segmentação com Mask R-CNN - {image_name}')
      plt.axis('off')
      plt.show()
```

Segmentação com Mask R-CNN - cachorro-preto-e-branco-1.png

Segmentação com Mask R-CNN - raca-de-cachorro-preto-e-branco2.jpg



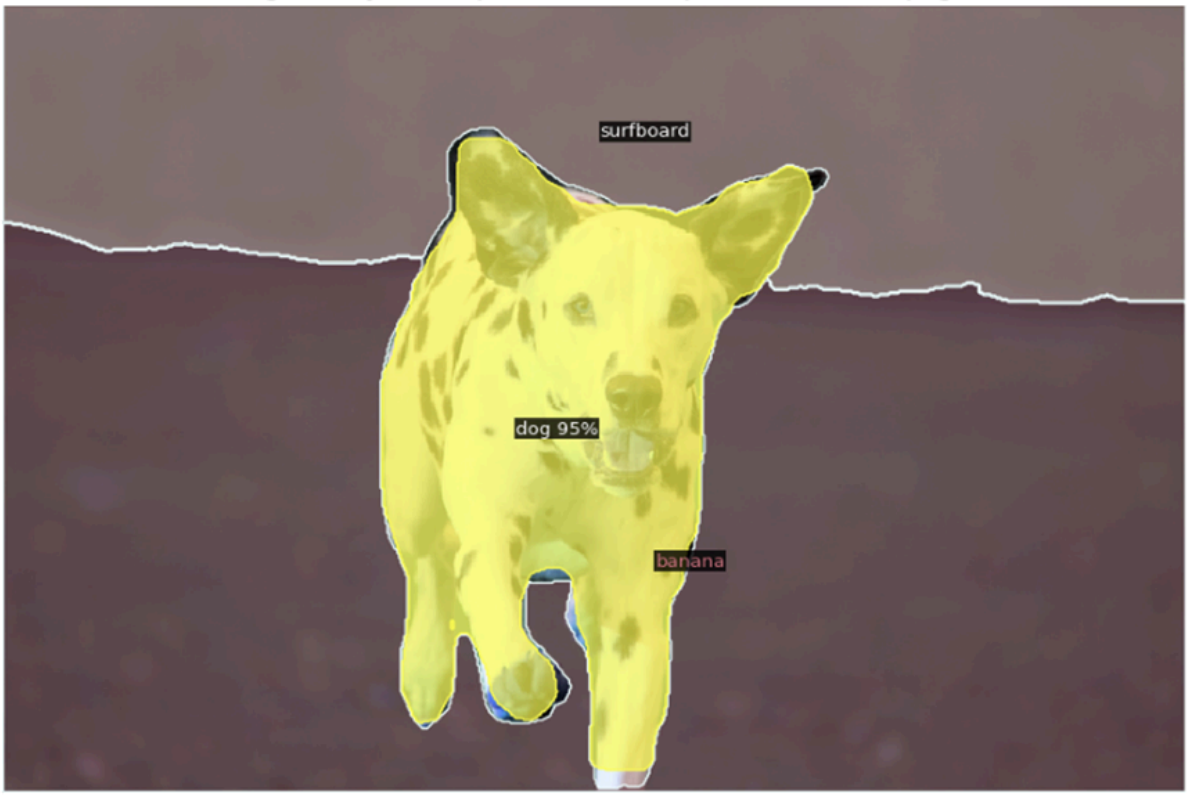
Segmentação Panóptica

```
 # Verifique se o dataset já está registrado
 dataset_name = "coco_2017_train_panoptic"

 if dataset_name not in DatasetCatalog.list():
      # Caminho para o dataset COCO Panoptic
      panoptic_root = "/path/to/coco/panoptic_train2017"  # caminho para
as imagens panópticas
      panoptic_json = "/path/to/coco/annotations/panoptic_train2017.json"
# caminho para o arquivo JSON de anotações
      image_root = "/path/to/coco/train2017"  # caminho para as imagens
originais
      metadata = {}  # metadados que serão utilizados

      # Registrar o dataset COCO Panoptic
     register_coco_panoptic(
```

```
        dataset_name,
      metadata=metadata,
      image_root=image_root,
        panoptic_root=panoptic_root,
        panoptic_json=panoptic_json
      )

 # Acessar os metadados do dataset
 MetadataCatalog.get(dataset_name)

 # Configuração do modelo Panoptic Segmentation no Detectron2
 def setup_panoptic():
      cfg = get_cfg()

cfg.merge_from_file(model_zoo.get_config_file("COCO-PanopticSegmentation/
panoptic_fpn_R_50_3x.yaml"))
     cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.5  # Definir o threshold
     cfg.MODEL.WEIGHTS =
model_zoo.get_checkpoint_url("COCO-PanopticSegmentation/panoptic_fpn_R_50
_3x.yaml")
     cfg.MODEL.DEVICE = "cuda" if torch.cuda.is_available() else "cpu"
      return cfg, DefaultPredictor(cfg)

 # Inicializar o preditor de Panoptic Segmentation
 cfg, predictor = setup_panoptic()

 image_folder = '/content/TestesSeg'

 # Percorrer todas as imagens na pasta
 for image_name in os.listdir(image_folder):
      if image_name.endswith(".jpg") or image_name.endswith(".png") or
image_name.endswith(".jpeg"):
      image_path = os.path.join(image_folder, image_name)

      # Carregar a imagem
      image = cv2.imread(image_path)

      # Aplicar Panoptic Segmentation na imagem
      outputs = predictor(image)
```

```
      # Obter a segmentação panóptica e as informações dos segmentos
      panoptic_seg, segments_info = outputs["panoptic_seg"]

      # Visualizar os resultados
      v = Visualizer(image[:, :, ::-1],
MetadataCatalog.get(dataset_name), scale=1.2)
      v = v.draw_panoptic_seg(panoptic_seg.to("cpu"), segments_info)

      # Exibir a imagem segmentada com Panoptic Segmentation
         plt.figure(figsize=(10, 10))
         plt.imshow(v.get_image()[:, :, ::-1])
      plt.title(f'Segmentação Panóptica - {image_name}')
      plt.axis('off')
      plt.show()
```

Segmentação Panóptica - cachorro-preto-e-branco-1.png



Segmentação Panóptica - raca-de-cachorro-preto-e-branco2.jpg

APÊNDICE 3

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 16 de out. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o **Stage 05:**

Encontrei mais dois surveys que me interessaram:

Foundational Models Defining a New Era in Vision - A survey and Outlook.pdf
FOUNDATIONAL MODELS IN MEDICAL IMAGING - A COMPREHENSIVE SURVEY AND FUTURE …

Esses estudos, junto com o survey do **Stage 04**, destacam as principais características dos Modelos Fundacionais sendo:

- Treinamento em larga escala.
- Uso de auto-supervisão ou semi-supervisão.
- Versatilidade e capacidade de adaptação.
- Eliminação da necessidade de treinar modelos específicos para cada tarefa.
- Propriedades emergentes e aplicação em diversos domínios.

Resumo de Definições: Modelos Fundacionais

Durante essa pesquisa encontrei um github onde encontrei alguns exemplos do modelo SAM usando o google colab:

tutorial_quickstart.ipynb
tutorial_text_prompt_seg.ipynb
tutorial_point_prompt_seg.ipynb

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- Explorar os modelos descobertos através do Survey.
- A partir dos exemplos encontrados na área médica, realizar testes em outros domínios usando o SAM.

- Definir outros modelos para testar nos domínios em que o SAM será avaliado.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

[Resumo de Definições:Modelos Fundacionais.doc citado no Termo de Aceite de Entrega de 16 de Outubro]

Resumo de Definições: Modelos Fundacionais

Image Segmentation in Foundation Model Era: A Survey

De acordo com o artigo, os **Modelos Fundacionais** (FMs) são definidos como modelos treinados em **dados amplos e variados**, frequentemente utilizando **auto-supervisão** em grande escala. Esses modelos são projetados para serem versáteis, permitindo sua adaptação para uma ampla gama de tarefas secundárias, eliminando a necessidade de treinar um modelo específico para cada tarefa.

O termo “Fundacional” reflete tanto a natureza central desses modelos para várias aplicações quanto sua capacidade de gerar **novas funcionalidades emergentes**. A proposta dos FMs é substituir múltiplos modelos específicos por uma abordagem mais genérica, criando uma base unificada para diferentes domínios e tarefas. Isso significa que, em vez de desenvolver modelos únicos para cada tarefa específica, os FMs oferecem uma estrutura única que pode ser adaptada para várias aplicações sem necessidade de extenso treinamento adicional ou ajustes.

Foundational Models Defining a New Era in Vision: A Survey and Outlook

De acordo com o artigo, os **Modelos Fundacionais** são definidos como **modelos baseados em dados em larga escala, treinados de maneira auto-supervisionada ou semi-supervisionada**, que podem ser adaptados para diversas tarefas secundárias. A importância destes modelos reside na substituição de vários modelos específicos por modelos mais amplos e genéricos, que, uma vez treinados, são facilmente ajustáveis para múltiplas aplicações.

Além de oferecer desenvolvimento rápido e bom desempenho em cenários variados, esses modelos também apresentam as chamadas **"propriedades emergentes"**, ou seja, comportamentos e capacidades que surgem de grandes modelos treinados em datasets massivos. Esses modelos estão transformando áreas como visão computacional e linguagem, permitindo uma maior generalização e contextualização das tarefas executadas,

e são amplamente aplicados em áreas como segmentação, reconhecimento de padrões, e até na interação multimodal envolvendo texto e imagem

Foundational Models in Medical Imaging: A Comprehensive Survey and Future Vision

De acordo com o artigo, os **Modelos Fundacionais** são definidos como modelos treinados em grandes conjuntos de dados utilizando técnicas de **auto-supervisão** ou **semi-supervisão**. Esses modelos servem como uma base genérica e flexível para uma ampla gama de tarefas secundárias (downstream), eliminando a necessidade de re-treino completo e reduzindo a dependência de dados rotulados em grande escala.

O artigo destaca que, ao contrário dos modelos tradicionais focados em tarefas específicas, os modelos fundacionais oferecem **generalização e adaptação** para diferentes modalidades, como textos médicos, registros eletrônicos de saúde e imagens médicas (como radiografias e exames de ressonância). Essa capacidade de adaptação é fortalecida por sua habilidade de aprender representações profundas e relevantes durante a fase de pré-treinamento, permitindo a aplicação eficiente em diversos cenários clínicos e de diagnóstico, com um mínimo de ajustes adicionais.

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 31 de out. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o **Stage 06:**

Encontrei um Artigo bem pequeno que listava alguns modelos:
High-Performance Few-Shot Segmentation with Foundation Models An Empirical Study.pdf

Utilizei este artigo para escolher os modelos que vou passar a testar a partir do próximo Stage, documentei eles junto com os Frameworks que eu tinha listado no Stage 04.
FrameworksStage06

Nesse documento atualizado tem alguns modelos que eu pretendo testar nas próximas semanas:

- DINOv2
- DFN (Data Filtering Network)
- CLIP
- SigLIP

Continuei buscando por testes do SAM, trouxe alguns dos colabs que achei mais interessantes:

video_predictor_example.ipynb
image_predictor_example.ipynb
how-to-segment-anything-with-sam.ipynb
FastSAM_example.ipynb
automatic_mask_generator_example.ipynb

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- Testar o DinoV2 para os mesmos exemplos que foram disponibilizados pela META nos colabs do SAM.
- Descobrir a viabilidade do uso desses modelos listados em segmentação de vídeos.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Em análise!

[FrameworksStage06.doc citado no Termo de Aceite de Entrega de 31 de Outubro]

1. PyTorch (Torchvision)

O PyTorch é uma biblioteca open-source de aprendizado profundo desenvolvida pelo Facebook AI Research. É amplamente utilizada por sua flexibilidade e facilidade de uso, especialmente para tarefas envolvendo processamento de imagens, visão computacional e processamento de linguagem natural. **Torchvision** é um pacote complementar ao PyTorch que fornece datasets, modelos pré-treinados e transformações para a visão computacional.

Documentação: Torchvision

2. OpenCV

OpenCV é uma biblioteca open-source usada para processamento de imagens e vídeos em tempo real. Ela é amplamente utilizada em projetos de visão computacional, desde operações básicas como manipulação de pixels até tarefas avançadas como detecção de objetos e rastreamento.

Documentação: OpenCV

3. Scikit-Image

Scikit-Image é uma coleção de algoritmos para processamento de imagens em Python, construída sobre as bibliotecas SciPy e NumPy. É especialmente útil para tarefas de segmentação, transformação, análise e manipulação de imagens, e é amplamente usada em pesquisa e desenvolvimento.

Documentação: Scikit-Image

4. Detectron2

Detectron2 é um framework de aprendizado profundo criado pelo Facebook AI Research para facilitar a implementação de modelos de detecção e segmentação de

objetos. Ele fornece uma coleção de modelos de última geração, como Faster R-CNN, Mask R-CNN, e Panoptic Segmentation.

Documentação: Detectron2

5. DINOv2

DINOv2 é um modelo auto supervisionado que aprende características visuais discriminativas sem a necessidade de anotações manuais. Ele é construído sobre a arquitetura Transformer e é eficaz na extração de representações robustas que podem ser aplicadas diretamente em tarefas de segmentação.

- **Principais Características**: DINOv2 gera mapas de ativação precisos que ajudam a localizar objetos em imagens, mesmo em cenários desafiadores com pouca supervisão.
- **Aplicações**: DINOv2 pode ser usado como *backbone* em frameworks de segmentação que requerem precisão na identificação e demarcação de regiões de interesse.

6. DFN (Data Filtering Network)

DFN é um modelo fundacional que melhora a precisão ao filtrar ruídos de grandes conjuntos de dados não organizados. Ele é especialmente útil quando combinado com outros modelos visuais, como DINOv2, para tarefas que exigem a fusão de conhecimento visual e linguístico.

- **Principais Características**: DFN introduz uma rede de filtragem de dados para melhorar a representação das características, tornando-o adequado para segmentação baseada em múltiplas entradas (por exemplo, texto e imagem).
- **Aplicações**: É frequentemente utilizado em cenários que combinam informações textuais e visuais, como a anotação de imagens com rótulos semânticos.

7. CLIP

CLIP (Contrastive Language-Image Pre-training) é um modelo treinado para associar texto e imagem, fornecendo uma base poderosa para tarefas de visão e linguagem. Embora não

seja um segmentador direto, CLIP pode ser usado para guiar a segmentação ao proporcionar uma compreensão contextual da imagem.

- **Principais Características**: CLIP pode alinhar embeddings visuais e textuais, permitindo que informações semânticas sejam incorporadas na segmentação.
- **Aplicações**: Ideal para segmentação orientada por linguagem, onde descrições textuais podem melhorar a precisão da identificação de objetos.

8. SigLIP

SigLIP é uma variação do CLIP que usa perda sigmoidal para treinar com lotes maiores de dados, melhorando as representações aprendidas. Ele é otimizado para tarefas que exigem alta precisão e minimização de ruído.

- **Principais Características**: Escala bem em cenários de treinamento massivo, proporcionando melhores representações semânticas.
- **Aplicações**: Usado em projetos que necessitam de integração robusta de visão e linguagem, com um foco em melhorar a segmentação semântica.

9. DeepLabv3

DeepLabv3 é uma arquitetura avançada de segmentação semântica que utiliza convoluções dilatadas (dilated convolutions) para capturar informações em diferentes escalas. Ela é especialmente eficaz em tarefas de segmentação semântica de alta resolução devido à sua capacidade de preservar detalhes espaciais em imagens.

- **Principais Características**: Oferece uma abordagem baseada em Redes Neurais Convolucionais (CNNs) com foco em capturar características contextuais e espaciais.
- **Aplicações**: Usado principalmente em segmentação semântica de alta precisão, como detecção de objetos em mapas ou imagens médicas.

10. CLIPSeg

CLIPSeg é uma variante do modelo CLIP, projetada especificamente para tarefas de segmentação. Ele aproveita a integração de visão e linguagem para gerar máscaras de segmentação baseadas em prompts textuais, oferecendo uma abordagem flexível e intuitiva para segmentação semântica.

- **Principais Características**: Permite segmentar imagens usando descrições textuais, com resultados robustos em cenários variados.

11. Transformers

Transformers é uma biblioteca open-source desenvolvida pela Hugging Face, que fornece uma coleção abrangente de modelos pré-treinados para diversas tarefas de visão computacional e processamento de linguagem natural, incluindo segmentação de imagens. Ela oferece suporte a modelos como CLIPSeg e Vision Transformers (ViTs), permitindo fácil integração em projetos.

- **Principais Características**: Suporte para modelos de visão e linguagem, flexibilidade para transfer learning e fácil integração com PyTorch.

APÊNDICE 4

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 7 de nov. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o **Stage 07:**

Iniciei o Stage tentando usar os modelos DINOv2 e CLIPSeg puros para segmentar imagens, mesmo que de forma simples:

"Segmentando" apenas com CLIP.ipynb
"Segmentando" apenas com DinoV2

Após essas tentativas, percebi que as segmentações não apresentavam bons resultados e passei a realizar um fine-tuning com o DINOv2:

FineTuningDinoV2.ipynb

Depois de concluir esses testes, comecei a pesquisar sobre segmentação de vídeo utilizando o DINOv2 e encontrei um artigo relevante:

1st Place Winner of the 2024 Pixel-level Video Understanding in the Wild (CVPR'24 PVUW) Challeng…

Atualizei a lista de frameworks com alguns que utilizei ou pretendo utilizar nos próximos **Stages**:
FrameworksStage07

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- Concluir o FineTuning do DinoV2
- Replicar em Escala Menor a Segmentação de Vídeo do Artigo para comparar com as segmentações de vídeo feitas pelo SAM2
  1st Place Winner of the 2024 Pixel-level Video Understanding in the Wild (CVPR'24 PVU…

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 13 de nov. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o **Stage 08:**

Comecei o Stage tentando resolver os problemas do Fine-Tuning do DinoV2, após algum tempo tentando acabei esbarrando nesse Blog: Link

Que produziu esse colab:

Train_a_linear_classifier_on_top_of_DINOv2_for_semantic_segmentation.ipynb

Depois de ter entendido um pouco melhor como funciona o treinamento eu decidi ir atrás de um Fine-Tuning do SAM para fins de comparação em um próximo Stage:

Fine_tune_SAM_(segment_anything)_on_a_custom_dataset.ipynb

E encontrei também esse ZeroShot do CLIPSeg:

Zero_shot_image_segmentation_with_CLIPSeg.ipynb

Durante esse processo também encontrei esse Blog falando sobre a junção de alguns modelos para uma segmentação melhor: Link

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- Continuar a pesquisar para implementar a segmentação de vídeo usando o CLIP e o DinoV2
- Escolher um dataset para poder comparar os testes a serem realizados com (CLIP, SAM2 e DinoV2) em imagens estáticas.
- Pesquisar um pouco mais sobre a junção desses modelos como descrito no blog.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

APÊNDICE 5

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 28 de nov. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o **Stage 09:**

Iniciei explorando a possibilidade de integração entre dois modelos fundacionais, o SAM e o GroundingDINO. Nesse processo, consegui adaptar uma implementação que demonstrou grande eficiência na combinação dos dois modelos para segmentação:

GroundingDINO-SAM

Percebi que poderia estar sendo pouco intencional em relação a algum resultado palpável e decidi iniciar testes a fim de comparar o uso dos modelos de segmentação:

U-Net com ResNet18
SAM sem Ajuste
SAM2 sem ajuste

Esses testes deram origem ao documento:

Análise Comparativa de Modelos de Segmentação – Stage 09

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- Finalizar os testes usando o SAM e o Dino em conjunto;
- Fazer ajustes finos nas duas versões do SAM para comparar antes e depois e também com os outros modelos;
- Verificar outras métricas que fazem sentido na avaliação de performance dos modelos selecionados;
- Testar todos os modelos selecionados em outros dois datasets (um deles na área da saúde).
- Finalizar o documento de Análise Comparativa de Modelos de Segmentação na sua "V2"

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

[Análise Comparativa de Modelos de Segmentação – Stage 09.doc citado no Termo de Aceite de Entrega de 28 de Novembro]

Análise Comparativa de Modelos de Segmentação – Stage 09

Nesta etapa do Stage, o foco principal foi realizar testes comparativos utilizando diferentes modelos de segmentação de imagens. Para isso, selecionei três modelos: U-Net com ResNet-18, SAM e SAM2. Cada modelo foi avaliado com base em suas características e contextos específicos de aplicação, e os resultados foram analisados com as métricas IOU (Intersection over Union) e Coeficiente de Dice.

Escolha dos Modelos

U-Net com ResNet-18
A U-Net foi escolhida como baseline devido à sua simplicidade e à comprovada eficácia em tarefas de segmentação sem a necessidade de grandes recursos computacionais. A utilização da ResNet-18 como backbone garantiu um bom equilíbrio entre desempenho e custo computacional. Por ser um método clássico, é ideal para servir como ponto de referência ao comparar com modelos mais avançados.

SAM (Segment Anything Model)
O SAM foi incluído por ser um modelo fundacional altamente estudado na literatura recente. Apesar de sua flexibilidade em segmentação geral, ele foi testado sem ajustes finos, o que nos permite avaliar seu desempenho inicial em um dataset específico. A inclusão desse modelo se justifica pelo alinhamento com o objetivo de explorar modelos fundacionais para tarefas específicas.

SAM2
A versão aprimorada do SAM, o SAM2, foi escolhida com o intuito de comparar a evolução entre as versões do modelo. Assim como o SAM, o SAM2 também foi testado sem ajustes finos, o que proporciona insights sobre como essas melhorias afetam o desempenho sem intervenções adicionais no treinamento.

Escolha das Métricas

As métricas IOU e Coeficiente de Dice foram selecionadas por sua relevância em tarefas de segmentação de imagens.

IOU (Intersection over Union):

Essa métrica mede a sobreposição entre a região predita e a região real, sendo amplamente usada em problemas de segmentação. Ela é robusta para avaliar quão bem o modelo consegue identificar as áreas de interesse e é especialmente útil em datasets com formas irregulares, como o Oxford-IIIT Pets.

Coeficiente de Dice:
Essa métrica é complementar ao IOU e é calculada com base na similaridade entre as áreas preditas e as reais. O Dice é sensível a pequenas discrepâncias e fornece uma visão mais detalhada do desempenho do modelo em regiões com bordas mais complexas. Sua inclusão garante uma análise mais abrangente e detalhada.

Essas métricas juntas fornecem uma visão completa do desempenho, desde a precisão global (IOU) até a qualidade de sobreposição das áreas segmentadas (Dice).

Resultados Obtidos

Modelo	IOU	Coeficiente de Dice
U-Net (ResNet-18)	0.8557	0.9167
SAM (sem ajuste)	0.4421	0.5635
SAM2 (sem ajuste)	0.4337	0.5564

Reflexões e Próximos Passos

Os testes realizados até o momento demonstraram a eficácia da U-Net como baseline, superando os resultados iniciais dos modelos fundacionais (SAM e SAM2) sem ajustes finos. Esses resultados destacam a necessidade de um trabalho mais direcionado para explorar todo o potencial dos modelos fundacionais.

Para as próximas etapas, as atividades serão focadas nos seguintes objetivos principais:

Finalizar os testes com a integração GroundingDINO-SAM

Consolidar os experimentos utilizando o SAM em conjunto com o GroundingDINO para avaliar a eficácia da integração em cenários específicos.

Realizar ajustes finos (fine-tuning) nos modelos SAM e SAM2

Ajustar ambos os modelos fundacionais ao dataset atual, permitindo comparações detalhadas entre:

- Desempenho antes e depois do fine-tuning.
- Resultados em relação aos outros modelos selecionados.
- Explorar métricas adicionais para avaliação de performance
- Identificar e aplicar métricas complementares que possam oferecer uma visão mais completa da performance dos modelos.

Ampliar os testes para novos datasets

- Testar os modelos selecionados (incluindo aqueles já implementados e os a serem desenvolvidos) em dois novos datasets:

- Um na área da saúde, para validar a aplicabilidade em um cenário crítico e relevante.
- Outro dataset, ainda em avaliação, possivelmente relacionado a imagens aéreas de drones ou câmeras de veículos autônomos.

Finalizar o documento comparativo – Versão 2

Incorporar os novos resultados no documento de análise comparativa, garantindo uma apresentação consolidada e robusta das conclusões obtidas.

Essas etapas permitirão não apenas fortalecer as análises realizadas até o momento, mas também explorar novas oportunidades no campo da segmentação, ampliando a aplicabilidade dos métodos avaliados.

[GroundingDINO-SAM.ipynb citado no Termo de Aceite de Entrega de 28 de Novembro]

```
# Clonar o repositório do GroundingDINO
!git clone https://github.com/IDEA-Research/GroundingDINO.git
!pip install -e ./GroundingDINO

# Instalar dependências específicas do projeto
```

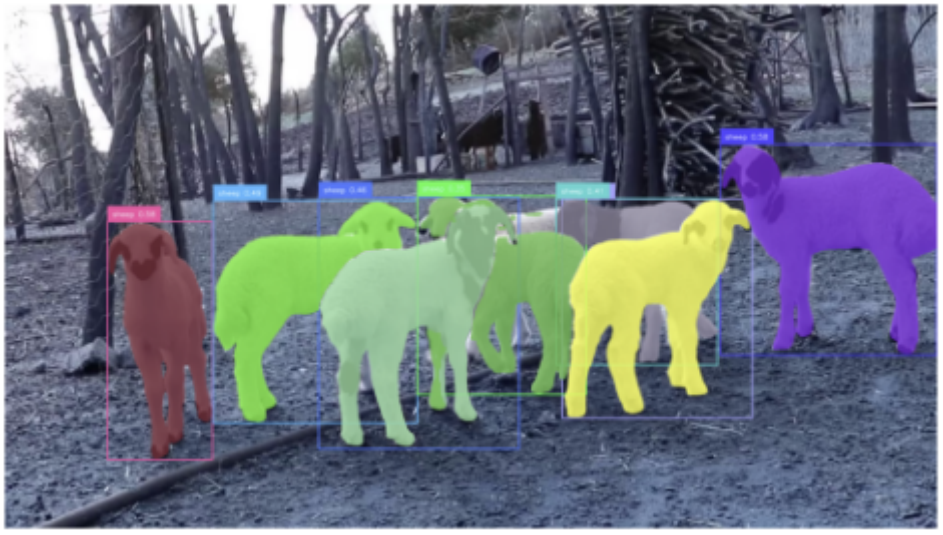
```
!pip install numpy opencv-python matplotlib torch torchvision
huggingface-hub pillow supervision ipython pyyaml tqdm
!pip install git+https://github.com/facebookresearch/segment-anything.git

# Baixar o checkpoint do modelo SAM
!wget
https://dl.fbaipublicfiles.com/segment_anything/sam_vit_h_4b8939.pth

import os
import sys
import cv2
import numpy as np
import torch
from PIL import Image
from huggingface_hub import hf_hub_download

# Adicionar o caminho do GroundingDINO ao sys.path
sys.path.append("./GroundingDINO")

# Importar os modelos do GroundingDINO e SAM
from GroundingDINO.groundingdino.models import build_model
from GroundingDINO.groundingdino.util import box_ops
from GroundingDINO.groundingdino.util.slconfig import SLConfig
from GroundingDINO.groundingdino.util.utils import clean_state_dict
from GroundingDINO.groundingdino.util.inference import annotate, predict,
load_image
from segment_anything import build_sam, SamPredictor

def load_dino_model_hf(repo_id, filename, ckpt_config_filename,
device='cpu'):
    # Baixa e carrega o modelo GroundingDINO
    cache_config_file = hf_hub_download(repo_id=repo_id,
filename=ckpt_config_filename)
    args = SLConfig.fromfile(cache_config_file)
    args.device = device
    model = build_model(args)

    cache_file = hf_hub_download(repo_id=repo_id, filename=filename)
    checkpoint = torch.load(cache_file, map_location=device)
    model.load_state_dict(clean_state_dict(checkpoint['model']),
```

```
strict=False)
    model.eval()
    return model

def grounding_sam(device):
    # Carrega os modelos GroundingDINO e SAM
    sam_checkpoint = 'sam_vit_h_4b8939.pth'
    predictor = sam_predictor(sam_checkpoint, device)

    ckpt_repo_id = "ShilongLiu/GroundingDINO"
    ckpt_filename = "groundingdino_swinb_cogcoor.pth"
    ckpt_config_filename = "GroundingDINO_SwinB.cfg.py"
    grounding = load_dino_model_hf(ckpt_repo_id, ckpt_filename,
ckpt_config_filename, device)

    return predictor, grounding

def sam_predictor(sam_checkpoint, device):
    # Cria o predictor do modelo SAM
    predictor =
SamPredictor(build_sam(checkpoint=sam_checkpoint).to(device))
    return predictor

def segment(image, sam_model, boxes, device):
    # Gera as máscaras a partir das boxes detectadas
    sam_model.set_image(image)
    H, W, _ = image.shape
    boxes_xyxy = box_ops.box_cxcywh_to_xyxy(boxes) * torch.Tensor([W, H,
W, H])

    transformed_boxes =
sam_model.transform.apply_boxes_torch(boxes_xyxy.to(device),
image.shape[:2])
    masks, _, _ = sam_model.predict_torch(
        point_coords=None,
        point_labels=None,
        boxes=transformed_boxes,
        multimask_output=False,
    )
    return masks.cpu()
```

```
def draw_mask(mask, image, random_color=True):
    # Desenha a máscara sobre a imagem
    if random_color:
        color = np.concatenate([np.random.random(3), np.array([0.8])],
axis=0)
    else:
        color = np.array([30 / 255, 144 / 255, 255 / 255, 0.6])
    h, w = mask.shape[-2:]
    mask_image = mask.reshape(h, w, 1) * color.reshape(1, 1, -1)

    annotated_frame_pil = Image.fromarray(image).convert("RGBA")
    mask_image_pil = Image.fromarray((mask_image.cpu().numpy() *
255).astype(np.uint8)).convert("RGBA")

    return np.array(Image.alpha_composite(annotated_frame_pil,
mask_image_pil))


def get_grounding_detect(image_path, text_prompt, model, box_threshold,
text_threshold):
    # Carrega a imagem como tensor e como fonte de entrada
    image_source, image_tensor = load_image(image_path)

    # Faz a predição das caixas, logits e frases baseadas no texto de
entrada
    boxes, logits, phrases = predict(
        model=model,
        image=image_tensor,
        caption=text_prompt,
        box_threshold=box_threshold,
        text_threshold=text_threshold,
    )

    # Adiciona anotações à imagem (caixas delimitadoras e descrições)
    annotated_frame = annotate(
        image_source=image_source, boxes=boxes, logits=logits,
phrases=phrases
    )
    annotated_frame = annotated_frame[..., ::-1]  # Converte de BGR para
```

```
RGB

    return annotated_frame, boxes


def image_segmentation(image_path, prompt, box_threshold, text_threshold,
predictor, model, device):
    # Carrega a imagem
    image_source, image_tensor = load_image(image_path)

    # Detecção de objetos e boxes
    annotate_image, detected_boxes = get_grounding_detect(image_path,
prompt, model, box_threshold, text_threshold)

    # Geração de máscaras
    masks = segment(image_source, predictor, boxes=detected_boxes,
device=device)

    # Desenha as máscaras sobre a imagem
    detection_with_mask = Image.fromarray(annotate_image).convert("RGBA")
    for i in range(masks.shape[0]):
        for j in range(masks.shape[1]):
            mask = masks[i][j]
            detection_with_mask = draw_mask(mask,
np.array(detection_with_mask), random_color=True)

    # Exibe e salva o resultado
    out_frame = cv2.cvtColor(np.array(detection_with_mask),
cv2.COLOR_BGR2RGB)
    plt.imshow(out_frame)
    plt.axis('off')
    plt.show()
    cv2.imwrite("segmented_image.jpeg", out_frame)


# Configuração e execução da segmentação de imagens
image_path = "/content/sheep1_0.jpg"  # Substitua pelo caminho correto no
Colab
prompt = "sheep"
box_threshold = 0.3
```

```
text_threshold = 0.25

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
predictor, grounding_model = grounding_sam(device)

# Executa a segmentação
image_segmentation(image_path, prompt, box_threshold, text_threshold,
predictor, grounding_model, device)
```



Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 4 de dez. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Gustavo Rodrigues da Silva

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

O **Stage 10** foi focado nos testes:

- UNet-ResNet18-Electron Microscopy.ipynb
- SAM-Electron Microscopy.ipynb
- SAM2-Electron Microscopy.ipynb
- SAM-fine tuning-Electron Microscopy.ipynb
- SAM2-fine tuning-Electron Microscopy.ipynb

- Oxford-IIIT Pet_FineTuning_SAM.ipynb

O conjunto desses testes foi descrito e comentado no documento:
Análise Comparativa de Modelos de Segmentação – Stage 10

Os próximos passos da minha jornada serão:

- Continuar buscando a segmentação utilizando Dino e SAM em conjunto
- Direcionar os estudos para a segmentação de vídeos utilizando os Modelos Fundacionais.

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

[Análise Comparativa de Modelos de Segmentação – Stage 10.doc citado no Termo de Aceite de Entrega de 04 de Dezembro]

Análise Comparativa de Modelos de Segmentação - Stage 10

Nesta etapa, o objetivo principal foi concluir os testes comparativos de segmentação de imagens utilizando diferentes modelos e explorar novas possibilidades com o fine-tuning de modelos fundacionais. Além de revisitar os modelos previamente utilizados no Stage anterior (U-Net com ResNet-18, SAM e SAM2), realizei ajustes nos modelos SAM e SAM2 para compará-los diretamente às suas versões sem ajuste, mantendo as métricas IOU e Coeficiente de Dice como padrão de avaliação devido à sua eficácia nos testes realizados.

Exploração de um Novo Dataset

O início do Stage foi direcionado à busca por novos datasets adequados para segmentação de imagens. Devido às limitações de recursos computacionais, encontrar um dataset que rodasse em tempo hábil foi um desafio. Durante esse processo, encontrei um tutorial 331 - Fine-tune Segment Anything Model (SAM) using custom data que apresentava um procedimento detalhado de fine-tuning do SAM utilizando o Electron Microscopy Dataset.

Esse dataset chamou atenção por sua aplicação na área da saúde, especialmente em patologia, e decidi utilizá-lo como segunda base de testes. No entanto, a segmentação eficaz desses dados mostrou-se bastante desafiadora devido à alta complexidade das imagens, o que exigiu um esforço significativo para ajustar e segmentar. Os resultados obtidos são apresentados na tabela abaixo:

Modelo	IOU	Coeficiente de Dice	Treinamento
UNet-ResNet1…	0.06	0.21	10 épocas
SAM-Electron …	0.52	0.61	Não treinado
SAM2-Electro…	0.53	0.62	Não treinado
SAM-fine tunin…	0.76	0.85	1 época
SAM2-fine tuni…	0.53	0.62	1 época

Desempenho Geral da U-Net (ResNet-18):

- A U-Net com ResNet-18, tradicionalmente utilizada como baseline em segmentação, apresentou os piores resultados, com um IOU de apenas **0.06** e um Coeficiente de Dice de **0.21**.
- Esses valores indicam que o modelo tem uma baixa capacidade de segmentar adequadamente as imagens do Electron Microscopy Dataset, o que pode ser atribuído à alta complexidade das imagens e à falta de especialização do modelo para esse tipo de dado.

Desempenho do SAM e SAM2 sem Fine-Tuning:

- Os modelos fundacionais, SAM e SAM2, tiveram um desempenho significativamente melhor que a U-Net mesmo sem fine-tuning. O IOU do SAM foi de **0.52**, e o SAM2 teve um IOU ligeiramente superior de **0.53**. O Coeficiente de Dice também reflete essa melhora, com valores de **0.61** e **0.62**, respectivamente.
- Esses resultados mostram que, mesmo sem ajustes, os modelos fundacionais têm uma maior capacidade de generalizar para datasets complexos, mas ainda não exploram todo o seu potencial.

Impacto do Fine-Tuning nos Resultados do SAM:

- O fine-tuning aplicado ao SAM resultou em uma melhora substancial nos seus valores de IOU (**0.76**) e Coeficiente de Dice (**0.85**). Esse aumento significativo demonstra que a adaptação do modelo ao dataset foi crucial para alcançar um desempenho superior.
- A personalização do SAM ao dataset de microscopia eletrônica permitiu que ele se tornasse muito mais eficaz na segmentação de imagens específicas.

Desempenho do SAM2 com Fine-Tuning:

- Surpreendentemente, o SAM2 com fine-tuning teve um desempenho equivalente ao SAM2 sem ajustes, com um IOU de **0.53** e um Coeficiente de Dice de **0.62**.
- Esse resultado pode indicar que o processo de fine-tuning não foi otimizado para o SAM2 ou que o modelo já atingiu seu limite de performance com esse dataset específico. Outra hipótese é que a arquitetura do SAM2 pode exigir mais refinamento ou dados adicionais para ajustar suas camadas fundacionais.

Revisitação ao Oxford-IIIT Pets

Após completar os testes no **Electron Microscopy Dataset**, retornei ao **Oxford-IIIT Pets** para aplicar o mesmo processo de fine-tuning aos modelos fundacionais. No entanto, durante os experimentos, enfrentei problemas técnicos com os dados, como inconsistências no carregamento e dificuldades na compatibilidade com os parâmetros do modelo. Esses problemas estão documentados no notebook relacionado, e como resultado, alguns valores ainda não foram calculados, sendo indicados com **"-"** na tabela abaixo:

Modelo	IOU	Coeficiente de Dice
Oxford-IIIT Pet_UNet-…	0.8557	0.9167
Oxford-IIIT Pet_SAM2…	0.4421	0.5635
Oxford-IIIT Pet_SAM.i…	0.4337	0.5564
Oxford-IIIT Pet_FineT…	-	-

Reflexões e Desafios

1. **Complexidade de Novos Datasets:**
   O **Electron Microscopy Dataset** trouxe um cenário desafiador, com imagens que exigem um alto nível de precisão. O fine-tuning provou ser essencial para melhorar os resultados, destacando a importância de personalizar os modelos para aplicações específicas.
2. **Problemas com o Oxford-IIIT Pets:**
   Apesar de ser um dataset menos complexo, dificuldades técnicas limitaram a execução completa dos testes. Esses problemas serão priorizados nas próximas etapas.

Próximos Passos

1. **Integração de DINO com SAM:**
   Continuar os estudos e testes para avaliar o potencial dessa integração em cenários mais avançados (para além do zeroshot).

2. **Segmentação de Vídeos:**
   Direcionar os esforços para expandir a análise de modelos fundacionais em segmentação de vídeos, investigando sua aplicação em cenários dinâmicos e mais desafiadores.

Com essas metas, pretendo aprofundar a pesquisa e aumentar a abrangência das aplicações dos modelos de segmentação, explorando desde imagens estáticas até vídeos complexos.

[SAM-fine tuning-Electron Microscopy.ipynb citado no Termo de Aceite de Entrega de 04 de Dezembro]

Instalando Bibliotecas Necessárias

```
%%capture

!pip install git+https://github.com/facebookresearch/segment-anything.git
!pip install -q git+https://github.com/huggingface/transformers.git
!pip install datasets
!pip install -q monai
```

Imports

```
import torch
from datasets import load_from_disk
from transformers import SamProcessor, SamModel
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
from tqdm import tqdm
from statistics import mean
from torch.optim import Adam
import monai
```

```
from torch.utils.data import Dataset, DataLoader
```

Carregando e preparando dataset

```
!gdown "1-GuYdpDqbV_QmfkOzuSP2ry9bXpNeuw5"
!gdown "1yEDhia3FutcTaeTsn3UI1ONrdG24Qt16"

!unzip -o "/content/train_dataset-20241204T073430Z-001.zip" -d
"/content/"
!unzip -o "/content/test_dataset-20241204T073428Z-001.zip" -d "/content/"

# Carregar os datasets pré-processados
train_dataset = load_from_disk("/content/train_dataset")
test_dataset = load_from_disk("/content/test_dataset")

# Inspecionar a estrutura dos datasets
print(f"Tamanho do train_dataset: {len(train_dataset)}")
print(f"Tamanho do test_dataset: {len(test_dataset)}")

# Inspecionar um exemplo do dataset
print(train_dataset[0])
print(test_dataset[0])

# Verificar as colunas disponíveis
print(f"Colunas do train_dataset: {train_dataset.column_names}")
print(f"Colunas do test_dataset: {test_dataset.column_names}")
```

Configurando modelo

```
%%capture

# Inicializar o processador e o modelo SAM
processor = SamProcessor.from_pretrained("facebook/sam-vit-base")
model = SamModel.from_pretrained("facebook/sam-vit-base")

# Congelar os encoders de visão e prompt, mantendo apenas o decoder
```

```
treinável
for name, param in model.named_parameters():
    if name.startswith("vision_encoder") or
name.startswith("prompt_encoder"):
        param.requires_grad = False

# Mover o modelo para o dispositivo
device = "cuda" if torch.cuda.is_available() else "cpu"
model.to(device)
# Função para obter bounding box a partir da máscara
def get_bounding_box(ground_truth_map):
    y_indices, x_indices = np.where(ground_truth_map > 0)
    if len(y_indices) == 0 or len(x_indices) == 0:  # Máscara vazia
        return [0, 0, 1, 1]  # Prompt mínimo
    x_min, x_max = np.min(x_indices), np.max(x_indices)
    y_min, y_max = np.min(y_indices), np.max(y_indices)
    return [x_min, y_min, x_max, y_max]

# Classe para dataset customizado
class SAMDataset(Dataset):
    def __init__(self, dataset, processor):
        self.dataset = dataset
        self.processor = processor

    def __len__(self):
        return len(self.dataset)

    def __getitem__(self, idx):
        item = self.dataset[idx]
        image = np.array(item["image"]) / 255.0  # Normalizar para [0, 1]
        if image.ndim == 2:  # Tons de cinza
            image = np.stack([image] * 3, axis=-1)  # Converter para RGB

        ground_truth_mask = np.array(item["label"])
        prompt = get_bounding_box(ground_truth_mask)
        inputs = self.processor(image, input_boxes=[[prompt]],
return_tensors="pt", do_rescale=False)
        inputs = {k: v.squeeze(0) for k, v in inputs.items()}
        inputs["ground_truth_mask"] = ground_truth_mask
        return inputs
```

```
# Criar datasets e dataloaders
train_dataset = SAMDataset(train_dataset, processor)
test_dataset = SAMDataset(test_dataset, processor)

train_dataloader = DataLoader(train_dataset, batch_size=2, shuffle=True,
drop_last=False)
```

Treinando o modelo

```
# Configurar otimizador e função de perda
optimizer = Adam(model.mask_decoder.parameters(), lr=1e-5,
weight_decay=0)
loss_fn = monai.losses.DiceCELoss(sigmoid=True, squared_pred=True,
reduction="mean")

# Loop de treinamento
num_epochs = 1
model.train()

for epoch in range(num_epochs):
    epoch_losses = []
    for batch in tqdm(train_dataloader, desc=f"Epoch
{epoch+1}/{num_epochs}"):
        optimizer.zero_grad()
        outputs = model(
            pixel_values=batch["pixel_values"].to(device),
            input_boxes=batch["input_boxes"].to(device),
            multimask_output=False
        )
        predicted_masks = outputs.pred_masks.squeeze(1)
        ground_truth_masks =
batch["ground_truth_mask"].float().to(device).unsqueeze(1)
        loss = loss_fn(predicted_masks, ground_truth_masks)
        loss.backward()
        optimizer.step()
        epoch_losses.append(loss.item())
    print(f"Epoch {epoch+1}: Mean Loss = {mean(epoch_losses):.4f}")
```

```
# Salvar o modelo treinado
torch.save(model.state_dict(), "sam_trained_model.pth")
```

Realizando as Inferências

```
# Certifique-se de que o modelo está no modo de avaliação
model.eval()

# Lista para armazenar as máscaras preditas
segmented_patches = []

# Loop pelas imagens do conjunto de teste
for idx in tqdm(range(len(test_dataset)), desc="Inferência no conjunto de teste"):
    item = test_dataset[idx]

    # Acessar os pixel_values da imagem (já processados)
    image = item["pixel_values"].cpu().numpy().transpose(1, 2, 0)  # De (C, H, W) para (H, W, C)

    # Acessar a máscara verdadeira para métricas
    true_mask = item["ground_truth_mask"]

    # Acessar o bounding box do prompt
    prompt = item["input_boxes"]

    # Preparar a imagem para o modelo (não precisa reprocessar, já está normalizada)
    inputs = {"pixel_values": item["pixel_values"].unsqueeze(0).to(device), "input_boxes": item["input_boxes"].unsqueeze(0).to(device)}

    # Inferência com o modelo
    with torch.no_grad():
        outputs = model(**inputs, multimask_output=False)

    # Aplicar sigmoide para converter logits em probabilidades
    prob_map = torch.sigmoid(outputs.pred_masks.squeeze(1)).cpu().numpy().squeeze()
```

```
    # Converter mapa de probabilidade para máscara binária
    threshold = 0.5  # Ajuste o limiar se necessário
    pred_mask = (prob_map > threshold).astype(np.uint8)

    # Adicionar a máscara predita à lista
    segmented_patches.append(pred_mask)

# Converter a lista de máscaras preditas em um array NumPy
segmented_patches = np.array(segmented_patches)

# Exibir informações sobre as máscaras segmentadas
print(f"Segmented patches shape: {segmented_patches.shape}")
```

Metrificando Resultados

```
# Funções para calcular IoU e Dice
def iou_score(y_true, y_pred):
    intersection = np.logical_and(y_true, y_pred).sum()
    union = np.logical_or(y_true, y_pred).sum()
    return intersection / union if union != 0 else 0.0

def dice_coefficient(y_true, y_pred):
    intersection = np.logical_and(y_true, y_pred).sum()
    return (2 * intersection) / (y_true.sum() + y_pred.sum()) if
(y_true.sum() + y_pred.sum()) != 0 else 0.0

# Listas para armazenar as métricas
iou_scores = []
dice_scores = []

# Loop para calcular as métricas
for idx in range(len(test_dataset)):
    # Acessar a máscara verdadeira e a predita
    true_mask = np.array(test_dataset[idx]["ground_truth_mask"])  #
Substituímos 'label' por 'ground_truth_mask'
    pred_mask = segmented_patches[idx]

    # Calcular IoU e Dice
```

```
    iou = iou_score(true_mask, pred_mask)
    dice = dice_coefficient(true_mask, pred_mask)

    # Adicionar as métricas às listas
    iou_scores.append(iou)
    dice_scores.append(dice)

# Calcular as médias das métricas
print(f"Mean IoU: {np.mean(iou_scores):.4f}")
print(f"Mean Dice Coefficient: {np.mean(dice_scores):.4f}")
```

Visualizando Resultados

```
# Visualizar os primeiros 5 exemplos do conjunto de teste
for idx in range(5):  # Ajuste o número de exemplos a serem exibidos, se necessário
    plt.figure(figsize=(12, 6))

    # Exibir a imagem original
    plt.subplot(1, 3, 1)
    image = test_dataset[idx]["pixel_values"].cpu().numpy().transpose(1, 2, 0)  # De (C, H, W) para (H, W, C)
    plt.imshow((image - image.min()) / (image.max() - image.min()), cmap="gray")  # Normalizar para [0, 1] para visualização
    plt.title("Imagem")

    # Exibir a máscara verdadeira
    plt.subplot(1, 3, 2)
    true_mask = test_dataset[idx]["ground_truth_mask"]
    plt.imshow(true_mask, cmap="gray")
    plt.title("Ground Truth")

    # Exibir a máscara predita
    plt.subplot(1, 3, 3)
    pred_mask = segmented_patches[idx]
    plt.imshow(pred_mask, cmap="gray")
    plt.title("Predição")

    plt.show()
```

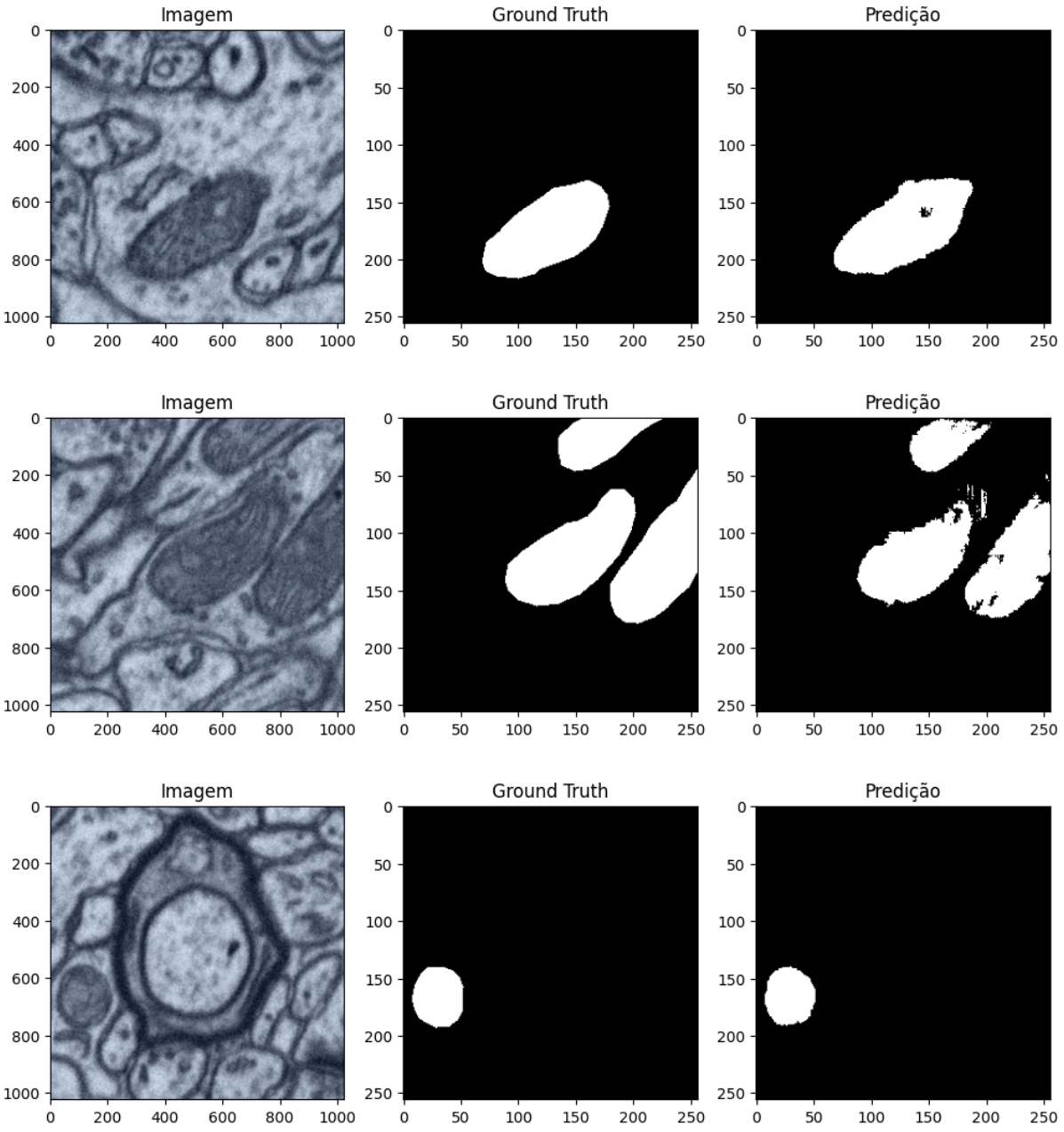


Imagem
Ground Truth
Predição
Imagem
Ground Truth
Predição
Imagem
Ground Truth
Predição