



UNIVERSIDADE ESTADUAL DE CAMPINAS

## Tarefa 2 - FLIM

MO445 - Análise de Imagem

Kristhian André Oliveira Aguilar (298976)

Nathan Shen Baldon (242448)

### Resumo

Este trabalho buscou unir os conhecimentos adquiridos durante o curso na tarefa de segmentação de ovos de *Schistosoma mansoni*. Como base, foi fornecido um *pipeline* de segmentação que utilizava um Perceptron Multicamadas (MLP), o algoritmo *Dynamic Trees* (DT) e um codificador FLIM. Com este *pipeline*, foram explorados diferentes hiperparâmetros e diferentes codificadores FLIM, obtendo-se ganhos superiores a 1% nas métricas F-beta, Dice e sensibilidade. Entretanto, frente a limitações do FLIM, optou-se pela exploração do modelo DinoV3 como possível substituto. Com isso, combinado ao algoritmo DT, foi possível melhorar significativamente as métricas **F-beta** (de 85,01% para **88,56%**) e **sensibilidade** (de 88,13% para **93,47%**), tal que esta se mostrou a melhor opção testada para o contexto médico.

Data de entrega: 07 de dezembro de 2025

## 1 Introdução

Na segunda parte do curso, dentre vários tópicos, estudamos formas de representar imagens, bem como formas de classificá-las e segmentá-las com redes neurais convolucionais (CNNs). Do ponto de vista didático, esta tarefa buscou a integração da primeira parte do curso, na qual estudamos segmentação por conectividade de *pixels*, com a segunda. Com isso, foi novamente utilizada delineação com o algoritmo de *Dynamic Image Foresting Transform* [1], mas aplicado a uma representação da imagem gerada com um Perceptron Multicamadas (MLP), e sementes geradas a partir do modelo de *Feature Learning from Image Markers* (FLIM) [2], capaz de aprender filtros a serem utilizados em CNNs.

Do ponto de vista de aplicação, o objetivo foi a delineação de ovos de *Schistosoma mansoni*, para futura contagem em amostras de fezes. Assim, o *dataset* utilizado foi de imagens de partes de lâminas com material contaminado, totalizando 1219 imagens (Fig.1).

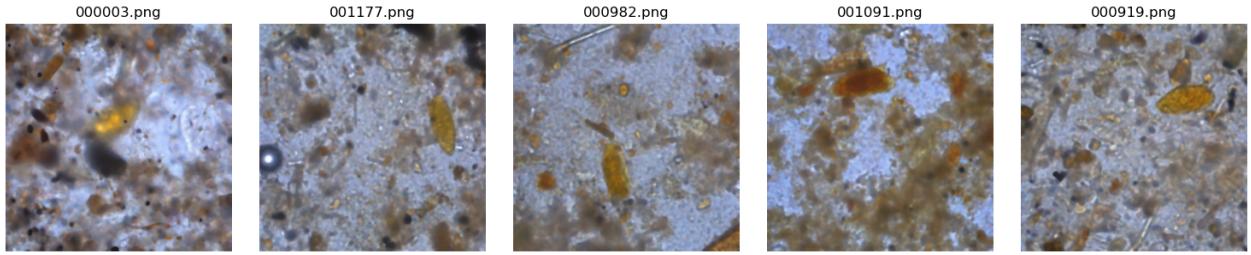


Figura 1: Cinco imagens aleatórias do *dataset* 'schisto'.

## 2 Metodologia

A tarefa foi abordada com dois *pipelines* diferentes: (i) o *pipeline* fornecido no roteiro da tarefa; (ii) e outro com o modelo DINOv3 ao invés do modelo FLIM. Ambos os *pipelines* estão apresentados nas seções seguintes, bem como as motivações por trás das mudanças realizadas.

### 2.1 Pipeline de Segmentação FLIM

O método original utilizado neste trabalho para segmentação de imagens combina uma abordagem do tipo *Image Foresting Transform* (IFT) baseada em marcadores — especificamente o algoritmo Dynamic Trees (DT) — com aprendizado profundo de representações. O fluxo de trabalho consiste em duas etapas distintas: (1) geração de um mapa de saliência para estimativa automática de marcadores utilizando um codificador FLIM e (2) estimativa de pesos de grafo aprendíveis por meio de um Perceptron Multicamadas (MLP) para guiar a delimitação realizada pelo IFT.

O método trata a segmentação de imagem como um problema de partição de grafo. Cada imagem é representada como um grafo, em que os *pixels* correspondem a nós e os arcos conectam *pixels* adjacentes. O peso de cada arco é definido pela dissimilaridade entre as representações de características (*features*) dos *pixels* conectados.

O pipeline é composto por três módulos principais. O codificador FLIM combinado a um decodificador adaptativo (Fig.2, (a)) gera um mapa de saliência aproximado para identificar a localização dos objetos. O estimador de pesos de grafo (MLP — Fig.2, (h)) extrai representações densas de

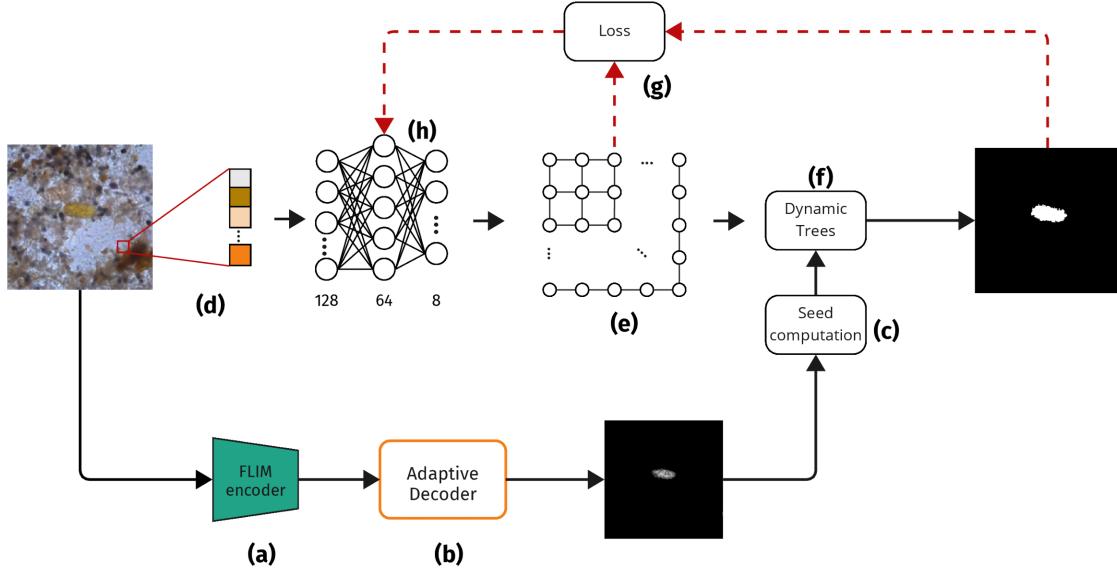


Figura 2: Arquitetura-base utilizada (retirado do roteiro da tarefa 2).

padrões locais (*patches*) da imagem, definindo assim a função de custo para o IFT. Finalmente, o módulo *Dynamic Trees* (DT — Fig.2, (f)) realiza a segmentação final propagando rótulos a partir dos marcadores, com base nos custos de caminho definidos pelas features do MLP.

Para automatizar a segmentação, são necessários marcadores iniciais (*seeds*). O codificador FLIM pré-treinado é utilizado para extrair características de alto nível da imagem de entrada. Esas características são processadas por um decodificador adaptativo, do tipo média com raio de adjacência de 1,5, gerando o mapa de saliência. Durante a inferência, esse mapa é binarizado utilizando o limiar de Otsu e filtrado pelo tamanho dos componentes conectados (mantendo componentes entre 1.000 e 9.000 *pixels*) para remover ruído. A máscara binária resultante é então erodida e dilatada para gerar marcadores internos (objeto) e externos (fundo), respectivamente.

O núcleo da etapa de delineação de contornos depende de uma rede neural denominada Estimador de Pesos de Grafo ou *Graph Weight Estimator* (GWE). Essa rede opera sobre pequenos *patches* da imagem, aprendendo um espaço *embedding pixel a pixel* em que a distância euclidiana entre vetores reflete as transições de fronteira do objeto. Para uma imagem de entrada  $I$ , são extraídos *patches* de tamanho  $k \times k$  (onde  $k = 7$  no método original) centrados em cada pixel. Os *patches* achados são então processados por um MLP composto por três camadas lineares: de  $3k^2 \rightarrow 128$  com função LeakyReLU, de  $128 \rightarrow 64$  também com LeakyReLU e, por fim, de  $64 \rightarrow D$ , onde  $D = 8$  é a dimensão de embedding. O resultado é um mapa de características de dimensão  $H \times W \times D$ , sendo que cada vetor  $v_p$  correspondente a um pixel  $p$  é normalizado por  $L_2$ . O peso do arco entre dois *pixels* adjacentes  $p$  e  $q$  é então definido como a distância euclidiana entre seus embeddings:  $w(p, q) = \|v_p - v_q\|_2$ .

Durante o treinamento, busca-se otimizar as representações do MLP para maximizar a precisão de delineamento obtida pelo algoritmo *Dynamic Trees*. As sementes FLIM são substituídas por “sementes oráculo” derivadas das máscaras de *Ground Truth* (GT), o que permite separar os erros de geração de marcadores dos erros de delineamento. As regiões do GT são erodidas e dilatadas para criar sementes internas ( $S_{in}$ ) e externas ( $S_{out}$ ), respectivamente. Em cada iteração de treinamento, o algoritmo *Dynamic Trees* é executado sobre as embeddings do MLP utilizando essas sementes oráculo, produzindo uma máscara predita  $M_{pred}$ . O treinamento segue um regime de aprendizado

contrastivo com *Triplet Margin Loss* [5], aplicada a *pixels* classificados corretamente pelo DT. Um *pixel* âncora (A) é comparado a um positivo (P) do mesmo rótulo e a um negativo (N) do rótulo oposto, aplicando-se o seguinte critério de margem  $\alpha = 0.2$ :

$$\mathcal{L} = \max(0, \|A - P\|_2 - \|A - N\|_2 + \alpha)$$

Esse mecanismo reforça a consistência das *features* dentro das regiões que já estão sendo corretamente segmentadas, enquanto realça o contraste nos limites dos objetos. Regularização adicional é aplicada aos parâmetros do MLP por meio de decaimento  $L_2$  dos pesos.

Na fase de inferência, o processo é executado passo a passo: a imagem é normalizada e processada pelo modelo FLIM para gerar o mapa de saliência; a partir desse mapa, são obtidas as sementes por operações morfológicas; a imagem é então passada pelo MLP treinado, que gera os *embeddings* de cada *pixel*; finalmente, a função ‘SMansoniDelineation’ — uma função especializada do módulo *Dynamic Trees* — utiliza os *embeddings* como pesos de grafo e as sementes derivadas do FLIM para produzir a máscara de segmentação final.

## 2.2 Pipeline de Segmentação DINOv3

Durante nossos experimentos com o método original baseado em FLIM e IFT, observamos que a qualidade das segmentações estava fortemente vinculada à capacidade do modelo FLIM de gerar mapas de saliência precisos (descrito na próxima seção — ‘Experimentos Preliminares’). Em muitos casos, o modelo falhava em destacar corretamente os ovos de *Schistosoma mansoni*, especialmente quando havia presença de detritos semelhantes no fundo. Para superar essas limitações, propomos uma nova metodologia que substitui o modelo FLIM pelo DINOv3 [4], um modelo Vision Transformer (ViT) [3] auto-supervisionado, combinado a um classificador linear leve para segmentação semântica. Nós exploramos tanto a integração do DINOv3 com o método original, substituindo o FLIM, quanto uma abordagem puramente baseada em DINOv3 para segmentação direta.

## 2.3 Treinamento do Classificador Linear (Dense Linear Probing)

A estratégia de treinamento adotada fundamenta-se na capacidade do modelo DINOv3 de gerar representações densas linearmente separáveis sem a necessidade de ajuste fino (finetuning) dos pesos da rede neural. Conforme demonstrado pelos autores do DINOv3 na seção de “Dense Linear Probing”[4], o modelo, quando utilizado como um extrator de características congelado (frozen backbone), produz vetores de características robustos o suficiente para que um classificador linear simples atinja o estado da arte em tarefas de segmentação semântica. Essa abordagem reduz drasticamente o custo computacional e a necessidade de grandes volumes de dados anotados, evitando o overfitting comum ao treinar Vision Transformers em datasets pequenos como o utilizado neste trabalho.

Diferente do método FLIM, onde o treinamento envolve a otimização de margens em triplas de pixels, o treinamento aqui consiste no ajuste de uma Regressão Logística sobre os patches da imagem. O processo foi implementado utilizando a biblioteca `scikit-learn` e seguiu os seguintes passos, conforme detalhado no notebook de treinamento `dino_segment.ipynb`:

- 1. Quantização e Filtragem de Rótulos:** Um passo crucial implementado no código é a adaptação das máscaras de Ground Truth (pixel a pixel) para a grade de patches do DINOv3.

Utiliza-se um filtro de convolução fixa (`patch_quant_filter`) que calcula a média dos valores da máscara binária dentro de cada região de  $16 \times 16$  pixels.

Para garantir que o classificador aprenda características representativas e não ruidosas, realizamos uma filtragem rigorosa dos dados de treino: apenas patches contendo "pureza" de classe foram mantidos. Especificamente, um patch é rotulado como fundo ( $y = 0$ ) se a média da máscara for  $< 0.01$  e como objeto ( $y = 1$ ) se for  $> 0.99$ . Patches de borda, que contêm uma mistura de pixels de fundo e objeto (valores entre 0.01 e 0.99), são descartados da matriz de design, pois suas características representam uma transição que poderia confundir o classificador linear.

2. **Extração de Características:** As imagens de entrada são redimensionadas para  $400 \times 400$  pixels e normalizadas. O modelo DINOv3, com tamanho de patch 16, processa a imagem e extrai as características da última camada. O resultado é um mapa de características de dimensão  $25 \times 25 \times 384$ , onde  $25 = 400/16$  e 384 é a dimensão do canal do modelo small. Ou seja, cada patch é representado por um vetor característico de dimensão 384.
3. **Otimização:** As características extraídas ( $X$ ) e os rótulos filtrados ( $y$ ) de todas as 326 imagens de treino são empilhados em grandes matrizes de dimensão ( $N \times 384$ ) e ( $N$ ) respectivamente, onde  $N$  é a quantidade de patches extraídos das imagens de treinamento, filtrados pelo passo 1 para manter apenas patches com rótulos de alta confiança. O classificador `LogisticRegression` é então ajustado utilizando o algoritmo de otimização L-BFGS com forte regularização ( $C = 1e-3$ ).

Não é utilizada retropropagação através do backbone do DINOv3. Isso permite que o treinamento do regressor converja em poucos segundos em uma CPU convencional ou GPU modesta, resultando em um arquivo leve de pesos (alguns kilobytes) que transforma as complexas representações do DINOv3 em mapas de probabilidade de segmentação.

## 2.4 Inferência

Durante a inferência, o pipeline processa as imagens de forma puramente feed-forward, permitindo processamento em batch inviável no algoritmo iterativo Dynamic Trees. A imagem de teste é processada pelo backbone, e, para cada patch na imagem, o classificador gera uma probabilidade do patch ser um ovo de parasita. O mapa de probabilidades resultante ( $H/16 \times W/16$ ) é interpolado por bilinearidade para a resolução original ( $400 \times 400$ ). Em seguida, é aplicada uma análise de componentes conexas (CCA) para garantir consistência topológica, descartando componentes menores que 400 pixels. Por fim, aplica-se uma dilatação morfológica à máscara predita, seguida de interseção lógica com um mapa de probabilidade de limiar inferior para refinamento das bordas. O processo descrito é aplicado tanto no método híbrido (DINOv3 + IFT) quanto no método puramente baseado em DINOv3.

Esse pipeline identifica de forma eficaz as instâncias corretas de ovos de *Schistosoma mansoni* em situações em que o modelo baseado em FLIM falha, especialmente em imagens com objetos de fundo semelhantes aos ovos.

## 2.5 Métricas de Avaliação

Para validar quantitativamente o desempenho dos métodos de segmentação propostos e comparar a eficiência entre as abordagens baseadas em FLIM e DINOv3, foram utilizadas as seguintes métricas:

---

- **Sensibilidade:** Avalia a proporção de pixels do objeto de interesse (ovos) que foram corretamente identificados pelo modelo em relação ao total de pixels que realmente compõem o objeto.
- **Especificidade:** Mede a capacidade do algoritmo de classificar corretamente as regiões de fundo, indicando a robustez do modelo em não confundir detritos ou ruído com a classe positiva.
- **F-beta Score:** Representa uma média harmônica ponderada entre precisão e sensibilidade, permitindo, através do parâmetro  $\beta$ , ajustar o peso dado à recuperação completa dos objetos em detrimento da precisão pura. Neste trabalho foi utilizado  $\beta = 3$ .
- **Tempo de Execução:** Mede o tempo necessário para realizar o processo de inferência no conjunto de teste.

## 3 Experimentos Preliminares

Nesta seção, são apresentados experimentos que antecederam a geração dos resultados finais discutidos na próxima seção, após a adoção do modelo DinoV3. Os experimentos preliminares foram realizados com o *pipeline* de segmentação FLIM descrito anteriormente, sendo divididos em três etapas: (1) exploração inicial; (2) novos treinamentos do FLIM; (3) adição de novo FLIM mais profundo (mais camadas).

### 3.1 Exploração Inicial

Com esta etapa, buscou-se visualizar as saídas de cada bloco e realizar pequenas variações nos hiperparâmetros do modelo. Primeiramente, foram exibidas as saída dos blocos GWE e FLIM (Fig.3).

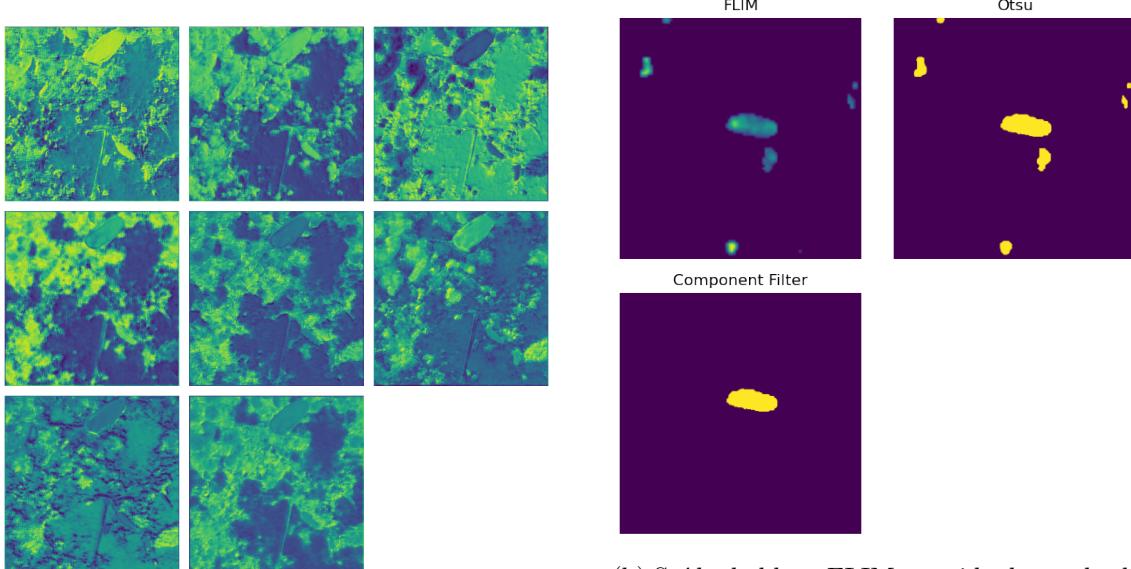
Com elas, foi possível analisar como a representação criada pelo bloco GWE é capaz de aumentar o contraste entre o ovo do parasita e sua vizinhança. Além disso, percebe-se o potencial do FLIM em ativar regiões dos ovos, bem como a importância da filtragem de componentes conexos, que elimina certas impurezas incorretamente ativadas.

Em seguida, foram testados diferentes funções peso de arco ( $arcw$  — 1 a 6 da Tarefa 1), porcentagem de sementes extraídas durante o treinamento (entre 0,5% e 50%), tamanho dos discos de erosão (5 a 15) e dilatação (10 a 40). A ideia dessas últimas variações (porcentagem de sementes e tamanho dos discos) foi deixar a quantidade de sementes mais escassas e distantes das bordas durante o treino, a fim de deixar o bloco GWE mais robusto a falhas do bloco FLIM.

A melhor função  $arcw$  foi a primeira ( $\|\mu_{R(p)} - I(q)\|$ ), semelhante aos resultados da Tarefa 1. Quanto às diferentes porcentagens de sementes e tamanhos dos discos, não houve nenhuma melhora significativa (nenhum aumento superior a 1%). No entanto, a representação criada pelo bloco GWE não se mostrou o principal gargalo para obtenção de um melhor desempenho: retirando erros inseridos pelo bloco FLIM, usando sementes do GT, era possível obter valores de F-beta superiores a 90% (utilizando discos e porcentagens citados anteriormente), tal que percebeu-se que era mais importante de se analisar os erros introduzidos pelo bloco FLIM.

Assim, optou-se por investigar as imagens do conjunto de treino que apresentavam piores resultados (F-beta abaixo de 20%), a fim de se identificar quais casos eram os mais difíceis para o modelo

---



(a) Oito canais da representação criada pelo bloco GWE.

(b) Saída do bloco FLIM, seguida do resultado da binarização e da filtragem de componentes conexos.

Figura 3

(Fig.4).

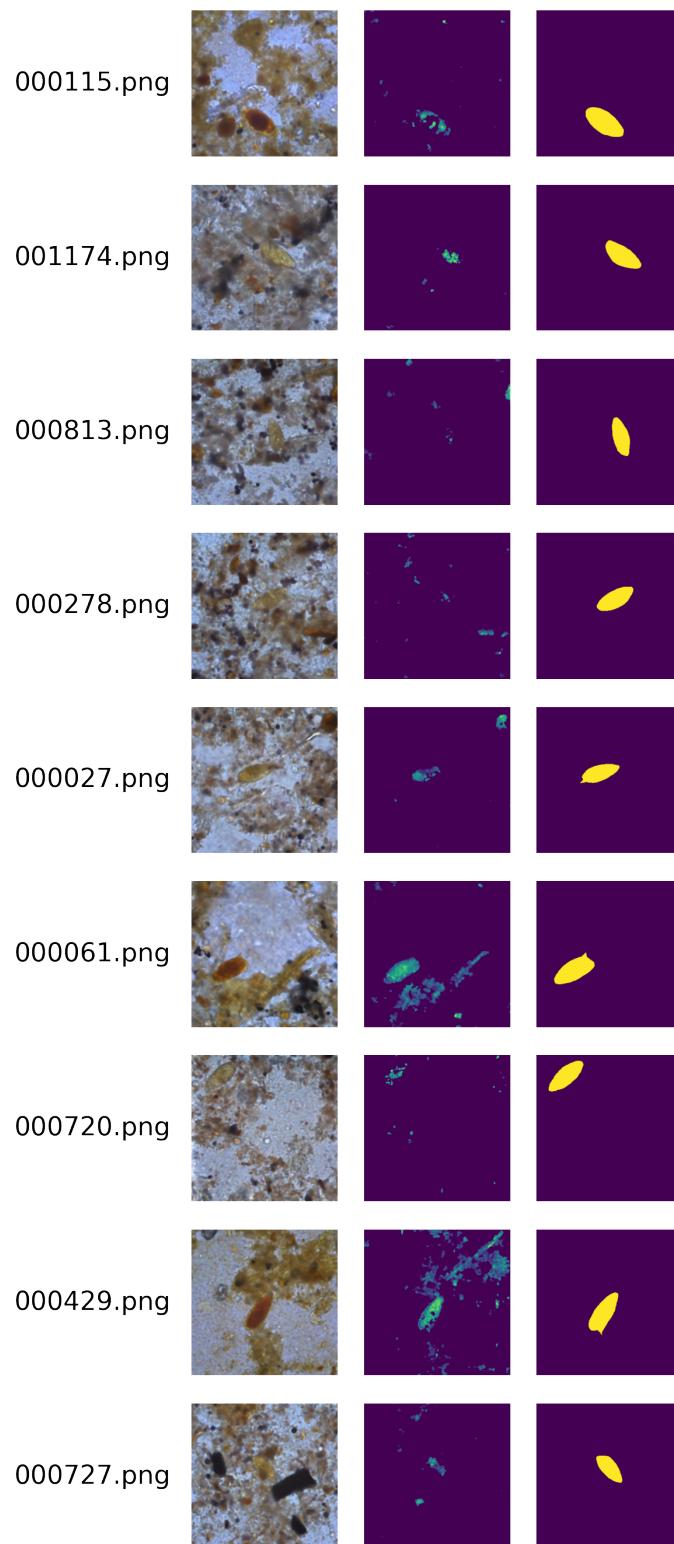


Figura 4: Imagens do conjunto de treino que tiveram valor de F-beta inferior a 20% após processamento pelo modelo *baseline*. A primeira coluna trata-se do nome da imagem, seguida da imagem original, do mapa de saliência gerado pelo bloco FLIM e pelo resultado eseprado.

---

No total, nove imagens obtiveram valor F-beta inferior a 20%. Analisando as três representações (imagem original, mapa de saliência e GT), observa-se que os piores casos podem ser divididos em duas categorias em que o FLIM apresenta maior dificuldade: **(i)** ovo do parasita claro (e.g., imagem '000278.png'); **(ii)** impurezas semelhantes à cor do ovo do parasita (e.g., '000429.png').

Observando as imagens utilizadas no treino do FLIM (Fig.5), pode-se observar que, de fato, nenhuma contém ovos mais claros (**caso difícil (i)**). Também, especificamente na imagem '000491.png', percebe-se que poderiam ser adicionadas sementes em estruturas de cor parecida com os ovos (**caso difícil (ii)**). Então, decidiu-se retreinar o FLIM tentando reduzir erros em ambos esses casos.

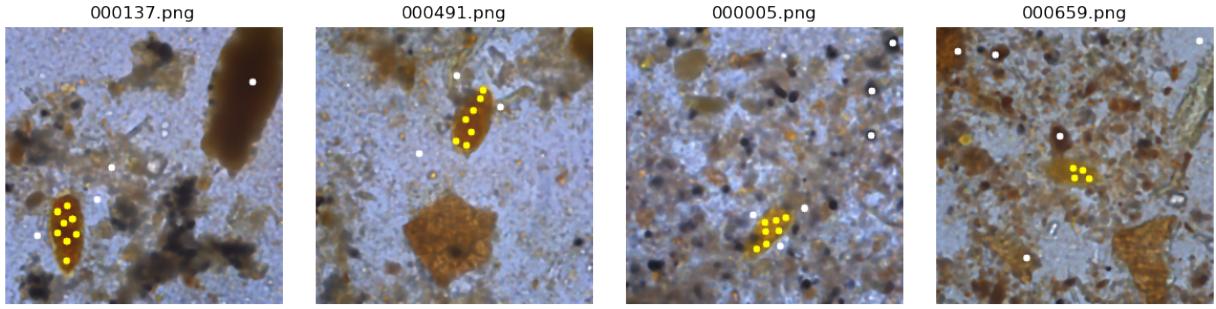


Figura 5: Imagens e marcadores utilizados no treino do FLIM *baseline*.

### 3.2 Novos Treinos do FLIM

Foram feitas duas tentativas de treino do FLIM, cada uma buscando melhorar o desempenho em um dos casos difíceis. Em ambos os casos, partiu-se das mesmas imagens e marcadores de treino do *split3*, uma vez que esse conjunto de imagens já apresentava bom desempenho. Para avaliar o resultado de cada treino, cada modelo treinado foi utilizado na segmentação do conjunto de validação ('val\_dataset\_gwe', mesmo do *notebook* fornecido). A avaliação foi feita com as métricas: F-beta, Dice, sensibilidade e especificidade. Os resultados obtidos foram apresentados na Tabela 1.

O primeiro novo conjunto de treino foi nomeado '*split4*' e consistiu apenas na adição de uma nova semente na imagem '000491/png' (Fig.6), selecionando um objeto de cor semelhante ao ovo. Em comparação com o FLIM pré-treinado no *split3*, apresentou pior desempenho, indicando que outro método deveria ser explorado para melhoria dos **casos difíceis (ii)**.

Já o segundo novo conjunto de dados ('*split5*', Fig.7) adicionou uma nova imagem ao *split3* ('000278.png'), que representaria melhor os **casos difíceis (i)**. Com essa mudança, houve aumento

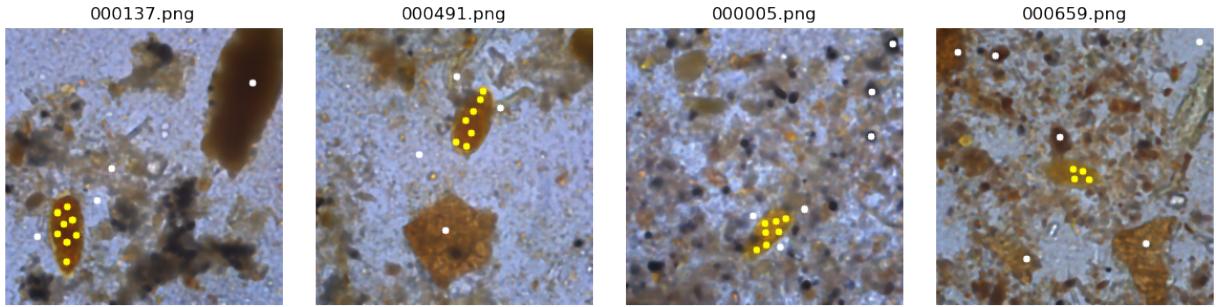


Figura 6: Imagens e marcadores do *split4* usados para novo treinamento do FLIM.

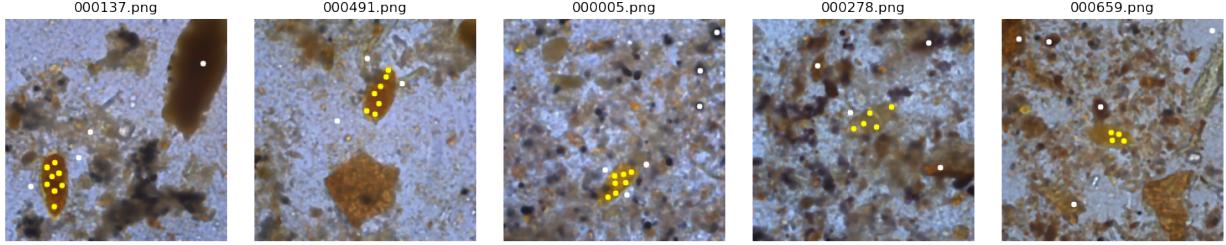


Figura 7: Imagens e marcadores do *split5* usados para novo treinamento do FLIM.

na métrica F-beta, reflexo do aumento na sensibilidade. No entanto, também houve maior ativação em regiões de impureza, o que reduziu significativamente a especificidade e o Dice, tal que esse método também não se mostrou interessante para aplicação no *pipeline* completo.

Tabela 1: Comparação entre diferentes métodos utilizados no bloco FLIM (diferentes conjuntos de treino). Métricas calculadas a partir do conjunto de validação.

Método	F-beta	Sensibilidade	Especificidade	Dice
Baseline (Split 3)	$0.6473 \pm 0.1515$	$0.6292 \pm 0.1505$	$0.9930 \pm 0.0146$	$0.6887 \pm 0.1782$
Baseline (Split 4)	$0.6261 \pm 0.1510$	$0.5910 \pm 0.1497$	$0.9958 \pm 0.0109$	$0.6851 \pm 0.1675$
Baseline (Split 5)	$0.6506 \pm 0.2337$	$0.7824 \pm 0.2365$	$0.9646 \pm 0.0359$	$0.5735 \pm 0.2543$

### 3.3 Adição do FLIM Profundo

De forma a explorar mais o **caso difícil (i)**, optou-se por uma nova abordagem, envolvendo o uso de dois modelos FLIM paralelos: o modelo FLIM *baseline*, e outro mais profundo, com maior capacidade de separação entre os diferentes *clusters* definidos pelos marcadores do conjunto de treino. Assim, enquanto o FLIM *baseline* teria mais informação de bordas (segmentação menos difusa), o FLIM mais profundo teria detecção mais precisa dos ovos (*clusters* mais separados).

Para unir os resultados dos dois blocos FLIM, foi utilizada a operação lógica *AND*, seguida da operação de reconstrução inferior implementada na biblioteca IFT (Fig.8). A primeira operação eliminaria regiões com falsos positivos e a segunda, recuperaria informações de borda obtidas pelo FLIM *baseline*.

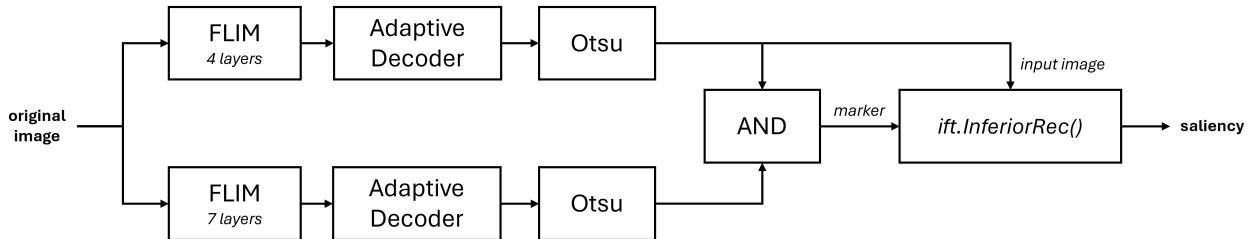


Figura 8: Novo *pipeline* do bloco FLIM com a adição de um FLIM profundo.

Foram testados FLIMs profundos com 6, 7, 8 e 12 camadas, sendo que o melhor foi o com **7 camadas**. Com essa implementação, novamente, não houve aumento expressivo em nenhuma métrica (Tab.2).

Tabela 2: Comparação entre FLIM baseline e novo pipeline com FLIM profundo. Métricas calculadas a partir do conjunto de validação.

Método	F-beta	Sensibilidade	Especificidade	Dice
Baseline (Split 3)	$0.6473 \pm 0.1515$	$0.6292 \pm 0.1505$	$0.9930 \pm 0.0146$	$0.6887 \pm 0.1782$
FLIM profundo (Split 3)	$0.6470 \pm 0.1573$	$0.6270 \pm 0.1570$	$0.9934 \pm 0.0143$	$0.6901 \pm 0.1782$

Um exemplo onde este método é interessante é apresentado na Figura 9.

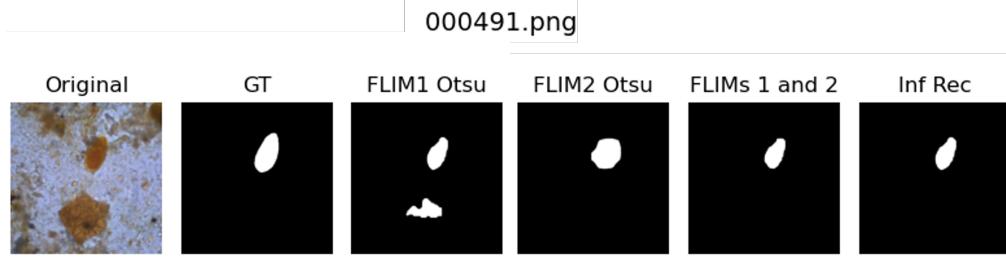


Figura 9: Exemplo das saídas de diferentes partes do *pipeline* com dois FLIMs (imagem '000491.png'). Da esquerda para a direita, as imagens se referem a: entrada ("Original"), resultado esperado ("GT"), saída do FLIM de quatro camadas após binarização ("FLIM2 Otsu"), saída do FLIM de sete camadas após binarização ("FLIM1 Otsu"), resultado da operação AND entre as duas imagens anteriores ("FLIMs 1 and 2") e saída da reconstrução inferior ("Inf Rec").

Neste, o FLIM *baseline* ('FLIM1 Otsu') possuía um componente conexo falso positivo, o qual não seria eliminado pela filtragem posterior. Enquanto isso, o FLIM mais profundo ativou apenas o ovo (FLIM2 Otsu), eliminando os falsos positivos. Neste caso, a necessidade da reconstrução inferior não fica explícita, uma vez que o resultado da operação AND é praticamente igual. Entretanto, há diversos outros casos em que o FLIM mais profundo (FLIM2) não ativa parte dos ovos, perdendo informações das bordas e sendo necessária a reconstrução inferior.

Como este método apresentou resultados interessantes ao ser avaliado isoladamente (apenas bloco FLIM), ele foi implementado no *pipeline* completo para avaliação no conjunto de teste, sendo analisadas as mesmas métricas anteriores (Tab.3). Note que houve aumento superior a 1% nas métricas F-beta, Dice e sensibilidade, além da redução dos desvios padrões.

Tabela 3: Comparação completa entre os métodos Baseline (FLIM + IFT) e Método com dois FLIMs. Métricas calculadas a partir do conjunto de teste.

Método	F-beta	Sensibilidade	Especificidade	Dice
<i>Métodos Originais (Baseline)</i>				
Baseline (FLIM + MLP)	$0.8501 \pm 0.3001$	$0.8813 \pm 0.3063$	$0.9931 \pm 0.0112$	$0.8215 \pm 0.2997$
Baseline (FLIM + Lab)	$0.8446 \pm 0.2886$	$0.8459 \pm 0.2886$	$0.9961 \pm 0.0097$	$0.8466 \pm 0.2919$
<i>Métodos com 2 FLIMs (Propostos)</i>				
2 FLIMs + MLP	<b><math>0.8646 \pm 0.2839</math></b>	$0.8946 \pm 0.2891$	$0.9937 \pm 0.0102$	$0.8370 \pm 0.2842$
2 FLIMs + Lab	$0.8546 \pm 0.2782$	$0.8546 \pm 0.2785$	$0.9966 \pm 0.0091$	$0.8575 \pm 0.2809$

Por fim, para analisar quantos dos piores casos foram resolvidos, foram comparados os casos com  $F - beta < 20\%$  entre o *pipeline baseline* e o *pipeline* com FLIM profundo. Os mesmos onze

piores casos ('000895.png', '001188.png', '000166.png', '000086.png', '000949.png', '000294.png', '001209.png', '000523.png', '000373.png', '000829.png', '001210.png') se repetiram em ambos os *pipelines*, indicando que, apesar de uma melhoria geral, não houve melhoria para os casos difíceis.

Frente aos resultados destes experimentos preliminares, optou-se pela exploração do *pipeline* de segmentação com o modelo DinoV3, descrito anteriormente. A seguir, são apresentados os resultados do *pipeline baseline* em comparação com o *pipeline* com o DinoV3.

## 4 Resultados e Discussão

A Tabela 4 mostra os resultados quantitativos obtidos na avaliação dos diferentes métodos de segmentação.

Inicialmente, fica claro que as abordagens que incorporam o DINOv3 são superiores às abordagens originais (Baseline FLIM) em praticamente todas as métricas de qualidade de segmentação. Tanto o método híbrido quanto a utilização exclusiva do DINOv3 apresentaram valores médios de F-beta e Dice mais elevados e desvios padrão menores, indicando não apenas uma melhor acurácia, mas também menor variabilidade. Isso corrobora a hipótese de que as features extraídas pelo Vision Transformer são superiores para diferenciar os ovos de parasitas do fundo complexo do que as features geradas pelo FLIM.

Ao comparar as variantes propostas entre si, observa-se que o método *DINOv3 Only* se destaca em termos de precisão e eficiência computacional. Este método obteve o maior F-beta score (0.8961) e um coeficiente DICE significativamente maior (0.8870) em comparação aos demais. Adicionalmente, em termos de custo computacional, o *DINOv3 Only* mostrou-se cerca de 40% mais rápido que o método híbrido de DINOv3 + MLP (61.18s contra 102.11s). Essa vantagem de velocidade decorre da eliminação da etapa iterativa e custosa do delineamento por grafo (IFT), realizando a segmentação de forma puramente *feed-forward*.

No entanto, o método simplificado possui uma clara limitação. O método híbrido *DINOv3 + MLP* apresentou uma sensibilidade significativamente maior (0.9347) do que a utilização isolada do DINOv3 (0.8766). Essa diferença evidencia o papel fundamental do refinamento de saliência promovido pelo algoritmo IFT (Dynamic Trees). Enquanto o classificador linear sobre o DINOv3 pode ser conservador nas bordas dos objetos, o IFT, guiado pelos pesos aprendidos pelo MLP, consegue propagar os rótulos de forma eficaz até os limites reais dos objetos, recuperando pixels que seriam perdidos pelo método simples.

Considerando o contexto da aplicação médica, onde a não detecção de partes do parasita (falso negativo) é geralmente mais penalizada do que a inclusão de pequenas regiões de fundo, a sensibilidade torna-se a métrica mais difícil e crítica de ser otimizada. Portanto, apesar do *DINOv3 Only* ser extremamente competente e rápido sozinho, o melhor modelo em questão de performance de detecção é o *DINOv3 + MLP*, pois combina a robustez de representação do transformador com a capacidade de delineamento fino do IFT, garantindo a recuperação mais completa dos objetos de interesse.

Tabela 4: Comparação completa entre os métodos Baseline (FLIM + IFT), Métodos Híbridos Propostos (DINOv3 + IFT) e DINOv3 Puro. Métricas calculadas a partir do conjunto de teste

Método	F-beta	Sensibilidade	Especificidade	Dice	Tempo (s)
<i>Métodos Originais (Baseline)</i>					
Baseline (FLIM + MLP)	$0.8501 \pm 0.3001$	$0.8813 \pm 0.3063$	$0.9931 \pm 0.0112$	$0.8215 \pm 0.2997$	94.16
Baseline (FLIM + Lab)	$0.8446 \pm 0.2886$	$0.8459 \pm 0.2886$	$0.9961 \pm 0.0097$	$0.8466 \pm 0.2919$	94.16
<i>Métodos Híbridos (Propostos)</i>					
Híbrido (DINOv3 + MLP)	$0.8856 \pm 0.2336$	<b><math>0.9347 \pm 0.2385</math></b>	$0.9930 \pm 0.0085$	$0.8416 \pm 0.2412$	102.11
Híbrido (DINOv3 + Lab)	$0.8860 \pm 0.2336$	$0.9172 \pm 0.2364$	$0.9948 \pm 0.0073$	$0.8569 \pm 0.2387$	102.57
<i>Método Simplificado</i>					
DINOv3 Only	<b><math>0.8961 \pm 0.2192</math></b>	$0.8766 \pm 0.2328$	<b><math>0.9983 \pm 0.0048</math></b>	<b><math>0.8870 \pm 0.2214</math></b>	<b>61.18</b>

### Análise de Robustez e Casos de Falha

Apesar das médias elevadas de desempenho, observa-se na Tabela 4 um desvio padrão considerável, especialmente na sensibilidade. A Figura 10 apresenta o gráfico de violino da sensibilidade para os modelos FLIM + MLP e DINOv3 + MLP.

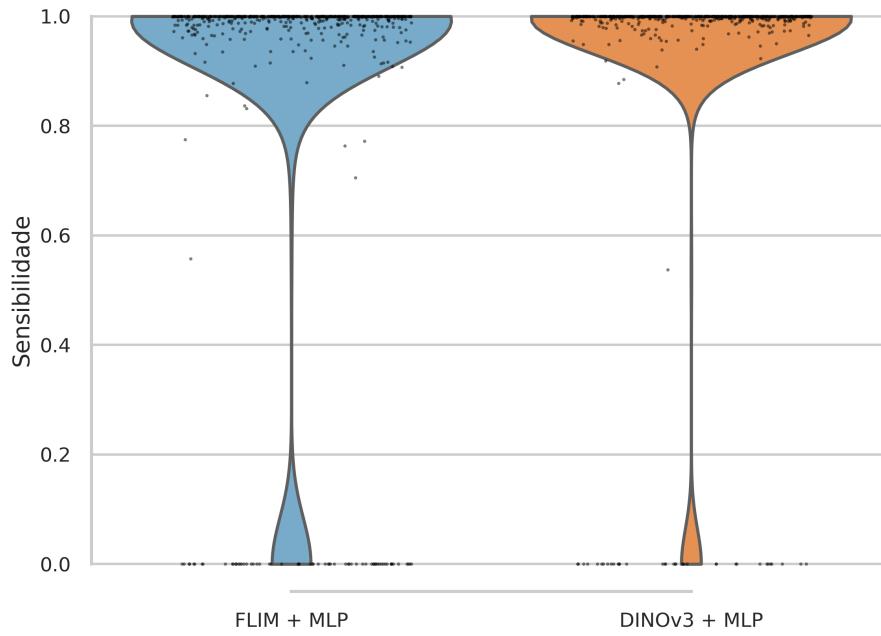


Figura 10: Distribuição da sensibilidade obtida pelos métodos FLIM + MLP e DINOv3 + MLP no conjunto de teste.

A distribuição evidencia que ambos os modelos acertam a grande maioria das previsões, concentrando a densidade de probabilidade próxima à sensibilidade 1.0 (topo do gráfico). No entanto, nota-se uma cauda longa estendendo-se até 0.0, indicando a ocorrência de erros catastróficos em um subconjunto específico de imagens. A análise quantitativa destes grupos reforça a superioridade da representação do DINOv3.

No conjunto de teste total de 610 imagens, o modelo **DINOv3 + MLP** obteve alta sensibilidade

( $> 0.8$ ) em 574 imagens (aprox. 94,1%), contra 541 imagens (aprox. 88,7%) do modelo **FLIM + MLP**. Mais crucialmente, o número de falhas graves (sensibilidade  $< 0.2$ ) foi drasticamente reduzido com o uso do DINOv3: apenas 35 imagens (5,7%) caíram nessa categoria, comparado a 64 imagens (10,5%) no modelo baseado em FLIM. Isso demonstra que a extração de características pelo DINOv3 é mais robusta e se confunde menos com objetos parecidos presentes no fundo.

Uma investigação detalhada das causas desses erros revela um viés introduzido pelo conjunto de dados. Como os conjuntos de treinamento e validação não continham exemplos de imagens sem ovos (máscaras vazias), os modelos desenvolveram um viés de expectativa de presença do objeto. Consequentemente, em imagens de teste contendo apenas detritos ou objetos morfológicamente similares aos ovos, os modelos tendem a alucinar previsões positivas.

Dos 35 casos de baixa performance do DINOv3, 19 (54%) correspondem a imagens onde a máscara de Ground Truth é vazia, indicando que a "falha" é, na verdade, uma alucinação (Falso Positivo). Para o FLIM, 26 dos 64 casos de falha (40%) são em máscaras vazias. Isso implica que, em imagens que realmente contêm ovos, o DINOv3 falhou em detectá-los apenas 16 vezes, enquanto o FLIM falhou 38 vezes, confirmando a maior capacidade de detecção do transformador.

A Figura 11 ilustra qualitativamente esses cenários de erro para o modelo DINOv3. Em aproximadamente metade dos casos mostrados (linhas 1 e 3), ocorre o erro de **alucinação**: a imagem não possui ovos (GT preto), mas o modelo identifica detritos ou artefatos minerais como positivos. Na outra metade (linhas 2, 4 e 5), ocorre o erro de **falso negativo** por baixa confiança: o ovo está presente, mas o modelo não consegue discerni-lo suficientemente de outros objetos semelhantes na imagem, resultando em uma pontuação baixa no mapa de probabilidade para o objeto real, o que leva à sua remoção durante a binarização. Ainda assim, a frequência reduzida desses erros no DINOv3 em comparação ao FLIM confirma sua maior aptidão para a tarefa.

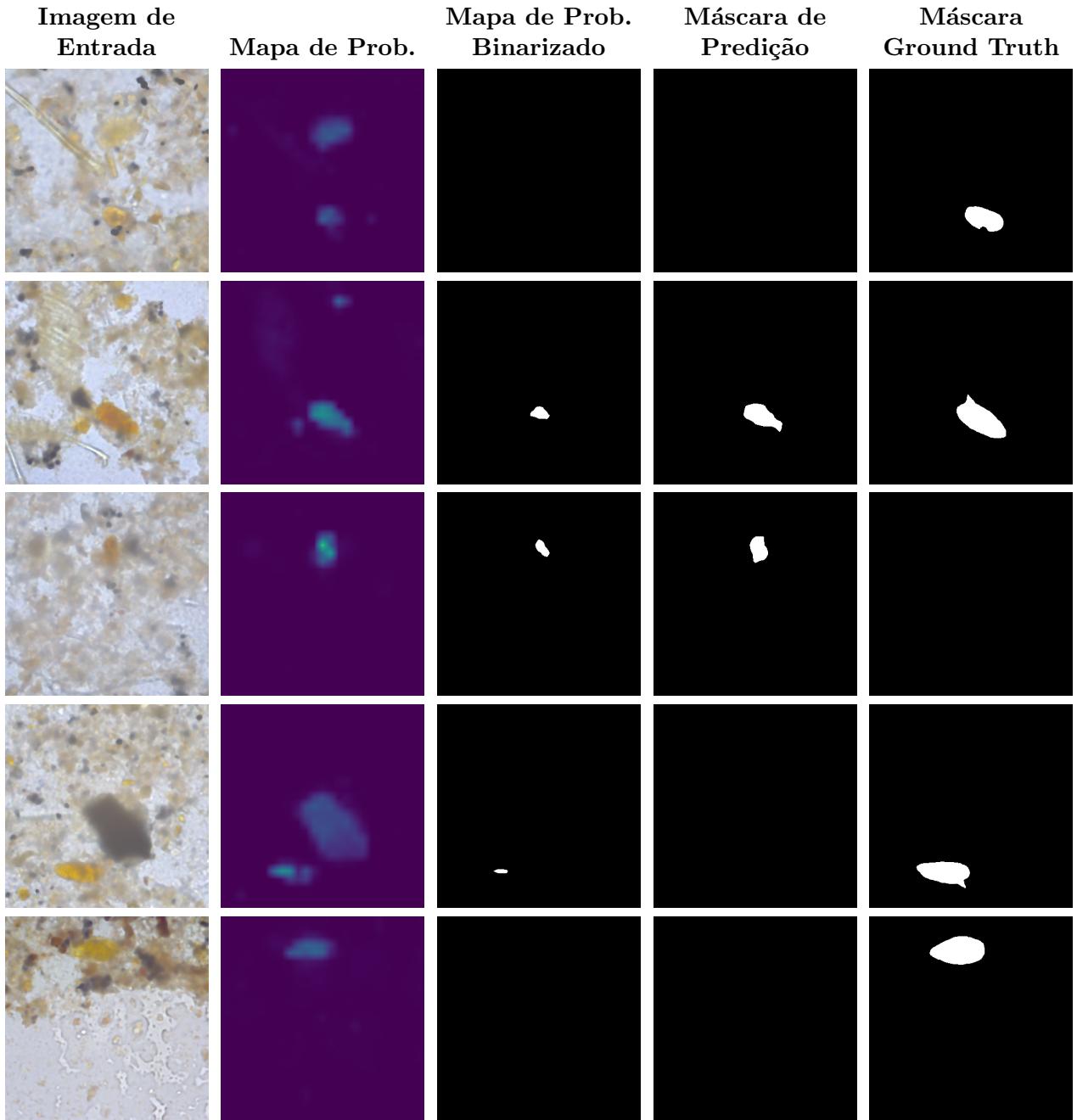


Figura 11: Resultados Qualitativos de imagens de baixa sensibilidade ( $<55$ ), obtidos pelo modelo DINOv3 no conjunto de teste. Da esquerda para a direita: Imagem de Entrada, Mapa de probabilidade, Mapa de probabilidade binarizado, Máscara de Predição e Máscara Ground Truth (GT).

## Conclusão

Com isso, do ponto de vista didático, foi possível abordar com sucesso diferentes partes do curso neste trabalho, conforme desejado. Do ponto de vista aplicado, foi possível aumentar o desempenho de segmentação dos ovos de parasitas em ambas as abordagens: *pipelines* com FLIM e DinoV3. No *pipeline* original, foram feitas variações em ambos os blocos GWE e FLIM, e, apesar de

aumentos superiores a 1% nas métricas F-beta, Dice e sensibilidade, foram encontradas limitações do bloco FLIM difíceis de serem superadas. Em especial, houve dificuldade em selecionar amostras representativas do *dataset* para treinamento do FLIM, de forma que sempre havia casos com erros frequentes deste bloco. Assim, optou-se pela exploração do DinoV3, o que proporcionou ganhos superiores a 3% nas métricas F-beta e Dice, além da redução dos desvios padrões de todas as métricas. Considerando o cenário médico, complementar a detecção do DinoV3 com a segmentação pelo IFT se mostrou a melhor opção testada, com sensibilidade muito superior aos outros *pipelines*.

## Referências

- [1] Jordão Bragantini, Samuel Botter Martins, Cesar Castelo-Fernandez, and Alexandre Xavier Falcão. Graph-based image segmentation using dynamic trees. In Ruben Vera-Rodriguez, Julian Fierrez, and Aythami Morales, editors, *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 470–478, Cham, 2019. Springer International Publishing.
- [2] Italos Estilon De Souza, Barbara C. Benato, and Alexandre Xavier Falcao. Feature Learning from Image Markers for Object Delineation. In 2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pages 116–123, Recife/Porto de Galinhas, Brazil, November 2020. IEEE. URL: <https://ieeexplore.ieee.org/document/9265976/>, <https://doi.org/10.1109/SIBGRAPI51738.2020.00024> doi:10.1109/SIBGRAPI51738.2020.00024.
- [3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL: <https://arxiv.org/abs/2010.11929>, <https://arxiv.org/abs/2010.11929> arXiv:2010.11929.
- [4] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprise, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3, 2025. URL: <https://arxiv.org/abs/2508.10104>, <https://arxiv.org/abs/2508.10104> arXiv:2508.10104.
- [5] Daniel Ponsa Vassileios Balntas, Edgar Riba and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 119.1–119.11. BMVA Press, September 2016. URL: <https://doi.org/10.5244/C.30.119>, <https://doi.org/10.5244/C.30.119> doi:10.5244/C.30.119.