Reconstructing Soybean Leaves: Geometric Modeling with Transformers

Alberto Yoshiriki Hisano Higuti[a], Eduardo Lopes de Lemos[a], Wesley Nunes Gonçalves[a]

[a]*Faculdade de Computação, Universidade Federal de Mato Grosso do Sul, Av. Costa e Silva, s/n, Campo Grande, 79070-900, MS, Brasil*

Abstract

Traditional vision models tend to emphasize texture over shape, which limits their ability to accurately segment the full shape of objects when they are occluded or degraded. This study introduces *Bormer*, a novel architecture we developed to prioritize shape inference over texture, addressing the challenges associated with object completion in partially occluded or degraded contexts. *Bormer* is evaluated against state-of-the-art language models, including T5-Small and LlaMA 3.2-1B, in a task centered on predicting defoliation in soybean leaves. Accurate leaf segmentation and reconstruction are critical in agriculture, where defoliation data can impact decision-making and productivity. Our results indicate that *Bormer* excels in reconstructing missing leaf segments, offering higher accuracy in defoliation analysis than competing models. This architecture demonstrates the potential of shape-focused models in agricultural and other applications where occlusion and degradation affect object visibility.

Keywords: Sketch Prediction, SeqtoSeq, Transformer, Computer Vision, Natural Language Processing, Soybean Leaf

Resumo

Modelos tradicionais de visão computacional tendem a enfatizar a textura em detrimento da forma, o que limita sua capacidade de segmentar com precisão o formato completo de objetos quando estão ocluídos ou degradados. Este estudo apresenta o *Bormer*, uma nova arquitetura desenvolvida para priorizar a inferência de formas em vez de texturas, enfrentando os desafios relacionados à reconstrução de objetos em contextos de oclusão parcial

ou degradação. O *Bormer* foi avaliado em comparação com modelos de linguagem de última geração, como T5-Small e LlaMA 3.2-1B, em uma tarefa focada na previsão de desfolha em folhas de soja. A segmentação e reconstrução precisas de folhas são cruciais na agricultura, onde os dados de desfolha podem impactar decisões e produtividade. Os resultados indicam que o *Bormer* supera os modelos concorrentes na reconstrução de segmentos ausentes de folhas, oferecendo maior precisão na análise de desfolha. Esta arquitetura demonstra o potencial de modelos centrados em formas para aplicações agrícolas e em outros contextos em que a oclusão e a degradação comprometem a visibilidade dos objetos.

Palavras-chave: Previsão de Esboços, SeqtoSeq, Transformer, Visão Computacional, Processamento de Linguagem Natural, Folha de Soja

---

1. Introdução

Nos últimos anos, os avanços em técnicas de *deep learning* têm proporcionado soluções robustas para problemas de segmentação em visão computacional. Modelos modernos de segmentação conseguem identificar e isolar objetos em imagens com alta precisão, com grande destaque na detecção de padrões visuais complexos (Chen et al., 2017; He et al., 2017). Esses modelos são eficazes em problemas onde as características visuais do objeto de interesse estão bem definidas e inteiramente visíveis. No entanto, eles apresentam limitações significativas quando confrontados com oclusões parciais ou quando a forma completa do objeto não está visível (Yuan et al., 2021; Purkait et al., 2019).

Essa limitação decorre do fato de que a maioria dos modelos de segmentação atuais é orientada por texturas e padrões de cor, sem uma compreensão mais profunda da geometria dos objetos (Ji et al., 2024; Aloun et al., 2022). Quando partes do objeto estão ausentes ou obscurecidas, os modelos não conseguem extrapolar ou prever as formas ocultas, deixando de lado informações importantes para uma segmentação completa (Yu and Wang, 2023). Para superar essa barreira, é necessário desenvolver modelos que sejam capazes de inferir a forma completa do objeto, mesmo em cenários com dados fragmentados. Neste contexto, técnicas que combinam visão computacional e processamento de texto emergem como soluções promissoras, capazes de integrar múltiplas fontes de informação para gerar predições mais precisas e confiáveis (Rinaldi et al., 2023).

Um dos modelos pioneiros a explorar o uso de técnicas de Processamento de Linguagem Natural (NLP) em tarefas de visão computacional foi o *Pix2Seq* (Chen et al., 2022), originalmente, proposto para detecção de objetos. O diferencial desse modelo reside no uso de princípios de modelagem de sequência, característicos de modelos de linguagem, para predizer as coordenadas dos objetos detectados. Em vez de utilizar âncoras de predição, o *Pix2Seq* modela a saída como uma sequência de *tokens*, e sinaliza o fim da predição quando todos os objetos na imagem foram identificados. Essa abordagem inovadora permitiu uma simplificação na arquitetura dos modelos de detecção ao eliminar a necessidade de âncoras fixas, usualmente empregadas em redes convolucionais.

A utilização dessa ideia inspirou o desenvolvimento de modelos mais avançados, como o *PolyFormer* (Liu et al., 2023), que combina modelos de linguagem e *text prompts* para realizar tanto a segmentação quanto a detecção de objetos em imagens. Baseado na arquitetura *Transformer* (Vaswani et al., 2017), o *PolyFormer* também trata a saída como uma sequência de *tokens*, em que cada um representa pontos geométricos da segmentação. A autoatenção presente nos *Transformers* permite ao modelo capturar dependências globais na imagem, e integrar informações de contexto local e global para segmentar objetos com alta precisão geométrica. Essa característica torna o *PolyFormer* uma solução robusta para aplicações onde a precisão na predição de contornos e formas é crucial, especialmente em cenários complexos.

Inspirado nessas abordagens, nós propomos o modelo *Bormer*, aplicando os princípios utilizados no *PolyFormer* e no *Pix2Seq* para solucionar uma tarefa similar. De forma diferente de seus predecessores, o *Bormer* não trabalha com imagens, mas diretamente com sequências de coordenadas. Essas coordenadas são tratadas como sequências de *tokens*, que configuram o modelo, essencialmente, como um modelo de linguagem. Isso faz com que nosso proposta possa ser aplicada diretamente na saída de qualquer modelo de segmentação. O *Bormer* é utilizado para completar formas deterioradas ou predizer áreas oclusas em objetos parcialmente visíveis, além de oferecer uma abordagem eficiente para a reconstrução de formatos geométricos complexos em diversas aplicações.

Essa abordagem é especialmente relevante em áreas como a agricultura de precisão, onde a análise correta de imagens de plantio pode influenciar diretamente a tomada de decisões e o aumento da produtividade (Kamilaris and Prenafeta-Boldú, 2018; Fracarolli et al., 2020). A adaptação dessas abordagens para o presente trabalho visa resolver um problema específico no

contexto agrícola: a identificação e completude de folhas de soja em imagens afetadas pela desfolha. Em muitas situações, devido a danos causados por pragas ou intempéries, as folhas de soja aparecem parcialmente removidas ou com grandes áreas danificadas, o que dificulta a análise correta do plantio (da Silva et al., 2019).

A proposta deste trabalho é desenvolver um modelo que, a partir de um vetor de coordenadas que representa os fragmentos das folhas já identificados, seja capaz de prever a continuidade dessas folhas, completar os segmentos faltantes ou corrigir distorções. Essa abordagem explorará a combinação de técnicas de visão computacional e predição sequencial para criar uma solução adaptada à problemática agrícola.

Além disso, modelos de linguagem atuais têm demonstrado um desempenho considerável quando ajustados para diferentes tarefas, principalmente após o fine-tuning. Ao considerar que a modelagem do problema proposto segue uma estrutura similar à de um problema de linguagem, este trabalho busca comparar o desempenho do modelo *Bormer* com modelos do estado da arte em tarefas gerais de linguagem, como o T5 *(Text-to-Text Transfer Transformer)* proposto pela Google Research, em 2019 (Raffel et al., 2019) e a recente família de LLMs, LlaMA *(Large Language Model Meta AI)* desenvolvida pela Meta AI, em 2023 (Touvron et al., 2023), especificamente, o LlaMA 3.2 lançada em 2024.

Além de sua aplicação direta no contexto agrícola, a abordagem proposta possui um grande potencial de aplicação em outras áreas. Por exemplo, ela pode ser utilizada no preenchimento de formas geométricas incompletas de objetos. A capacidade do modelo de identificar e completar padrões visuais, mesmo em cenários com dados incompletos ou ruidosos, demonstra sua flexibilidade e abrangência. Esse potencial para adaptar o modelo a diferentes domínios amplia as possibilidades de uso, o que torna ele uma ferramenta versátil tanto para o setor agrícola quanto para outras áreas que demandam a restauração de padrões visuais.

Portanto, este trabalho busca não apenas aplicar técnicas de *deep learning* no preenchimento de folhas de soja em imagens de plantio, mas também explorar os limites da combinação entre visão computacional e predição sequencial para criar um modelo capaz de atuar em múltiplos contextos. Ao adaptar conceitos do *PolyFormer* e do *Pix2Seq*, a solução aqui proposta poderá contribuir para uma análise mais precisa e eficiente em cenários onde a informação visual é fragmentada, e possibilitar novas aplicações em diversos setores.

2. Metodologia

Na Figura 1, foi apresentada uma visão geral da abordagem empregada neste trabalho por meio da utilização de um fluxograma, em que os blocos interiores dele serão explicados detalhadamente, nas subseções seguintes.

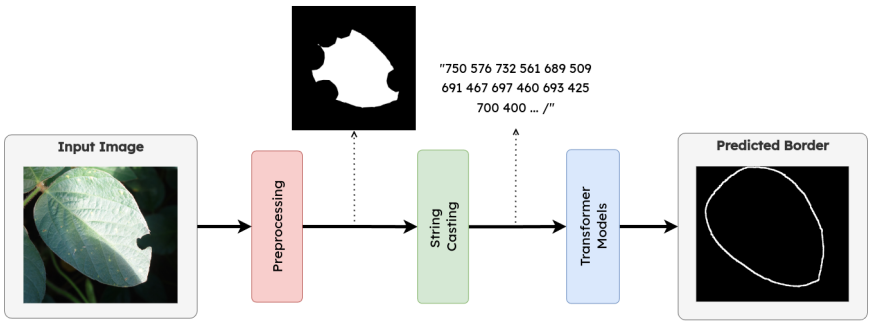

Figura 1: Fluxograma Principal do Experimento

2.1. Datasets

Os experimentos deste trabalho foram conduzidos com a utilização de um *dataset* composto por diversas fotografias de folhas de soja, capturadas em plantações de soja por dispositivos móveis, as quais foram obtidas a partir do PlantVillage (Hughes and Salathe, 2015). As imagens possuem diferentes resoluções e escalas, com uma variação de 367×367 pixels a 800×800 pixels e um total de 1089 imagens. As imagens das folhas de soja possuem 3 canais (R,G,B). Esse *dataset* já foi utilizado em estudos prévios (Bressan et al., 2022), onde foi fornecido uma divisão de 94% das imagens para treinamento e 6% para teste. Alguns exemplos das imagens utilizadas para o treinamento dos modelos podem ser observados na Figura 2.

Figura 2: Imagens do dataset original

As anotações deste dataset consistem de máscaras de segmentação, onde cada máscara é uma imagem binária que destaca as áreas de interesse, ou seja, as folhas e separa elas do que é considerado fundo da imagem.

2.2. Preprocessing

Para simular o processo de degradação foliar causado por intempéries naturais, foi implementado um processo de degradação sintética das máscaras de segmentação do dataset de folhas. Este processo não apenas enriquece o conjunto de dados com cenários desafiadores, mas também é indispensável para validar a robustez e a generalização dos modelos. Ele simula cenários onde as folhas aparecem com partes ausentes ou ocluídas, algo comum em aplicações reais devido a fatores como danos físicos, sombras ou sobreposições de outras folhas. Esse processo é realizado diretamente nas máscaras de segmentação do dataset de folhas, como ilustrado na Figura 3.

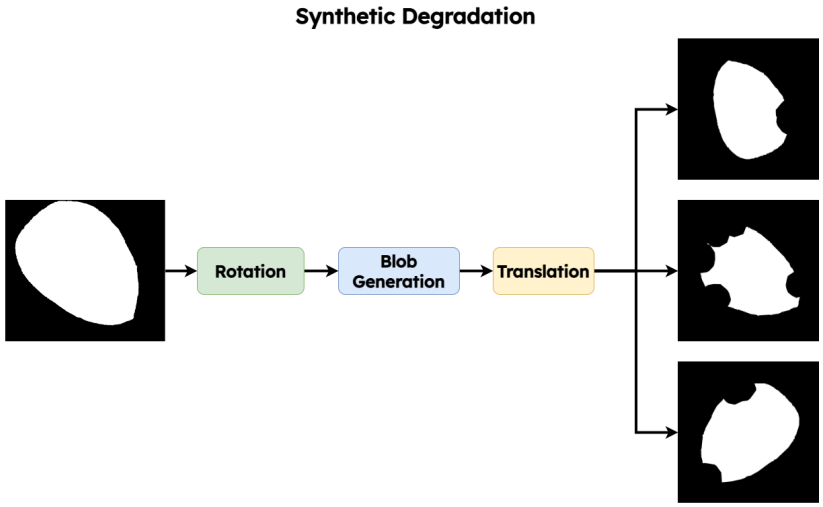

Figura 3: Processo de Degradação Sintética dos Dados

O processo de degradação de dados sintéticos segue uma série de passos guiados por elementos aleatórios para garantir uma variedade de cenários e mudanças nas formas de degradação. Primeiramente, extrai-se o vetor correspondente aos pontos que formam o contorno da máscara de segmentação da folha. Esse vetor é uma representação precisa da borda da folha segmentada, e cada ponto corresponde a uma coordenada específica ao longo desse contorno.

A rotação desses pontos por um ângulo aleatório, selecionado uniformemente no intervalo de 0° a 360°, desempenha um papel crucial nesse processo. Ela não apenas introduz variações geométricas que ampliam a diversidade dos cenários, mas também garante que o modelo seja exposto a padrões de degradação que ocorrem sob diferentes orientações no mundo real, como em folhas dobradas ou giradas por ação do vento.

Em um passo subsequente, um ou mais pontos ao longo do vetor de contorno são selecionados aleatoriamente. Para cada ponto escolhido, é gerado um *buffer* ao seu redor, cujo formato e tamanho são definidos de maneira aleatória. O formato do *buffer* pode variar entre figuras geométricas simples, como círculos e quadrados, ou formas mais complexas, conhecidas como *blobs*.

Um *blob* consiste na determinação de diversos pontos aleatórios ao redor de um ponto central, sendo que cada ponto criado possui um ângulo e um raio aleatório. O ângulo é definido dentro de um intervalo que varia entre 0 e 2π, enquanto o raio é determinado aleatoriamente entre 0 e o parâmetro *spread*. A quantidade de pontos que define o formato do *blob* é especificada pelo parâmetro *num_points*.

Por fim, uma translação aleatória é aplicada ao vetor de contorno com o objetivo de adicionar deslocamentos que simulam mudanças de perspectiva ou variações na posição relativa das folhas dentro da imagem. Essa translação, limitada a até 15% da largura total da imagem, permite explorar cenários em que as folhas aparecem parcialmente fora do enquadramento ou deslocadas devido ao movimento. Uma representação desse processo pode ser visualizado na Figura 4.

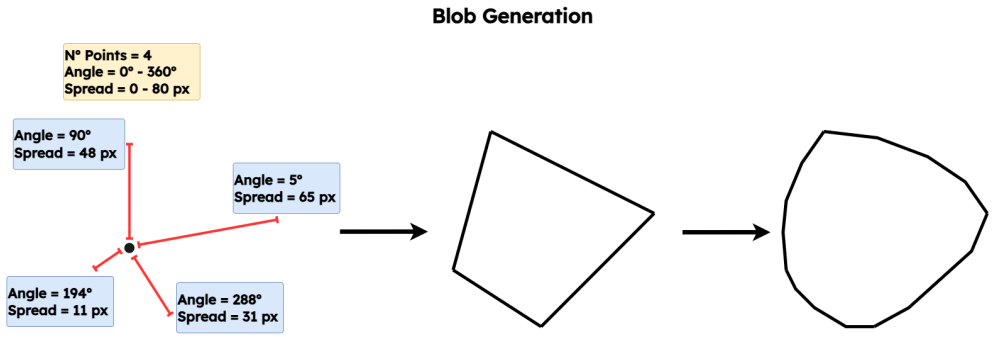

Figura 4: Processo de geração dos *blobs*

Adicionalmente, um *buffer* aleatório é aplicado sobre esses *blobs*, e pode variar entre os valores de *min_ radius* e *max_ radius*. Para a geração dos *blobs* utilizados neste trabalho, foram considerados os seguintes valores para os parâmetros:

- *spread*: 80 pixels
- *min_ radius*: 1 pixel
- *num_ points*: 200
- *max_ radius*: 30 pixels

É importante ressaltar que os valores desses parâmetros podem variar de acordo com o tamanho das imagens do *dataset* utilizado. Todos os códigos dos pré-processamentos aplicados ao *dataset* podem ser encontrados no repositório oficial[1] do projeto.

Uma vez gerado o *buffer*, este é sobreposto à máscara de segmentação original, e toda a área de interseção entre o *buffer* e a máscara é removida. Em outras palavras, os pixels contidos dentro da área de interseção são apagados para simular a ausência de parte da folha. Esse procedimento pode ser repetido várias vezes, utilizando diferentes pontos, tamanhos e formas de *buffers*, para criar múltiplas regiões degradadas em uma mesma máscara de segmentação.

Essa abordagem, além de ser fundamental para a criação de cenários realistas de degradação foliar, é essencial para testar a capacidade dos modelos de prever e completar as partes faltantes das folhas de maneira eficaz. Ao confrontar o modelo com uma ampla gama de situações possíveis, incluindo orientações, posições e formas variadas de degradação, o processo não apenas reforça sua robustez, mas também o prepara para lidar com desafios reais no campo, garantindo análises agrícolas mais precisas e confiáveis.

2.3. String Casting

Após o procedimento explicado na subseção 2.2, é necessário preparar as entradas de forma compatível com as arquiteturas utilizadas, então as anotações (máscaras) degradadas também devem estar representadas de forma vetorial ao extrair as bordas delas. Uma vez que as bordas são extraídas, aplica-se a técnica de simplificação de curvas Douglas-Peucker para reduzir a complexidade dos contornos.

[1] GitHub

A técnica Douglas-Peucker é um método iterativo que foi originalmente desenvolvido para simplificar a representação de curvas e linhas em dados espaciais, com o objetivo de reduzir o número de pontos que definem uma forma sem comprometer sua estrutura geral (Douglas and Peucker, 1973). Sua aplicação é extremamente útil em cenários onde os contornos são descritos por uma quantidade excessiva de pontos, tornando o processamento e a inferência computacionalmente mais custosos.

O princípio do funcionamento desse algoritmo pode ser descrito em algumas etapas:

1. Definição da Tolerância: Primeiramente, define-se um valor limite *epsilon*, representado na Figura 5 pela letra grega ϵ, que é a tolerância para o erro entre a curva simplificada e a curva original.
2. Identificação do Ponto Mais Distante: O algoritmo começa ao conectar o primeiro e o último ponto da curva (pontos a e b na Figura 5.1) original com uma linha reta (representada em linha verde e tracejada). Em seguida, encontra-se o ponto na curva original que está mais distante dessa linha reta (no ponto c da Figura 5.1).
3. Recursão: Se a distância desse ponto à linha for maior que ϵ, ele é mantido na representação simplificada, e o processo é repetido recursivamente para os segmentos da curva antes e depois desse ponto. Caso a distância seja menor, o ponto é descartado.
4. Simplificação da Curva: Esse processo é repetido até que todos os pontos tenham sido processados, para resultar em uma versão simplificada da curva com menos pontos, mas ainda preserva sua forma geral, como pode ser observado na Figura 5.3 (pela linha tracejada verde).

É fundamental destacar que o parâmetro ϵ, que define a tolerância de distância utilizada na simplificação das curvas, será sempre menor que o tamanho do *blob* gerado durante o processo de degradação sintética da folha, conforme descrito na subseção anterior. Essa precaução é necessária para evitar que a simplificação reduza os pontos analisados de forma iterativa a uma linha reta, comprometendo assim a preservação das características estruturais da folha.

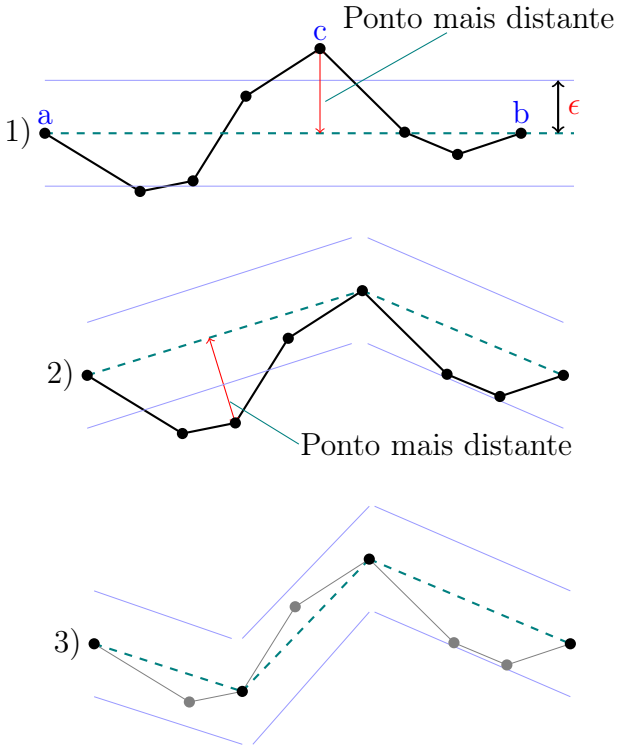

Figura 5: Fundamento do Algoritmo de Douglas-Peucker (Fonte:(Jeronymo, 2023))

Um exemplo para esse processo e aplicação da técnica Douglas-Peucker pode ser observada na figura 6.

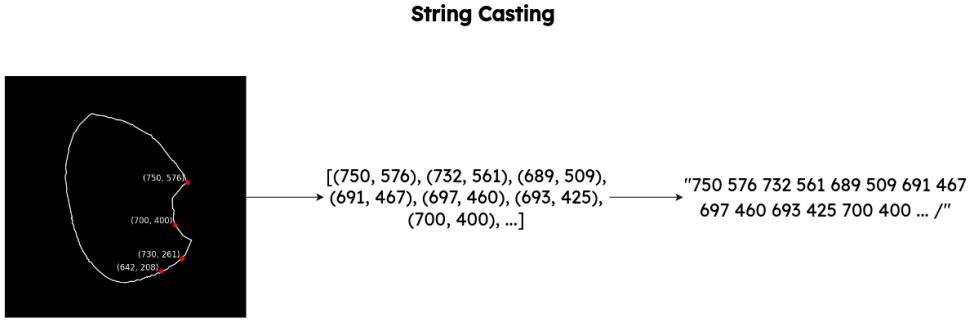

Figura 6: Processo usado para conversão em formas vetoriais

Com isso, obtém-se uma redução na quantidade de pontos de controle sem perder a forma geral da folha, o que facilita a inferência geométrica e permite que o modelo reconheça e complete as partes faltantes de maneira eficiente, ao utilizar a estrutura global da folha.

2.4. Models

2.4.1. T5-Small

O T5 (*Text-to-Text Transfer Transformer*) (Raffel et al., 2019), foi concebido como uma solução unificada para diversas tarefas de Processamento de Linguagem Natural (NLP), onde todas as tarefas são formuladas no formato texto-para-texto. Essa formulação permite ao modelo abordar de maneira consistente tarefas como classificação, tradução, e geração de texto, o que também o posiciona como um potencial candidato para tarefas de completude de formatos. O T5 utiliza a arquitetura *Transformer*, originalmente proposta por (Vaswani et al., 2017), e está disponível em diferentes tamanhos, para adequar-se a uma variedade de cenários computacionais, desde aplicações leves até tarefas que exigem maior capacidade de processamento.

Foi utilizada a versão pré-treinada do T5, disponibilizada em Hugging Face. A arquitetura do T5, baseada em mecanismos de atenção, mostrou-se compatível com o contexto das tarefas propostas, com evidências de bons resultados em diversos benchmarks. Entre as versões, a que ofereceu maior compatibilidade com a arquitetura comparada, o *Bormer*, foi o T5-Small, com aproximadamente 60 milhões de parâmetros.

2.4.2. LlaMA 3.2-1B

O LlaMA 3.2, parte da terceira geração de modelos da família LlaMA, representa uma linha de modelos de larga escala voltados para tarefas de Processamento de Linguagem Natural (NLP) (Touvron et al., 2023). Projetado para aprimorar o desempenho em tarefas de geração e compreensão de texto, o LlaMA 3.2 foi construído com o objetivo de fornecer resultados robustos em diversos cenários, e manter a eficiência computacional.

Assim como seus predecessores, o LlaMA 3.2 é baseado na arquitetura *Transformer*, que utiliza mecanismos de atenção para lidar com a interdependência de palavras em sequências textuais (Vaswani et al., 2017). Essa arquitetura é amplamente reconhecida por seu sucesso em tarefas de NLP, como tradução automática, sumarização de texto e resposta a perguntas.

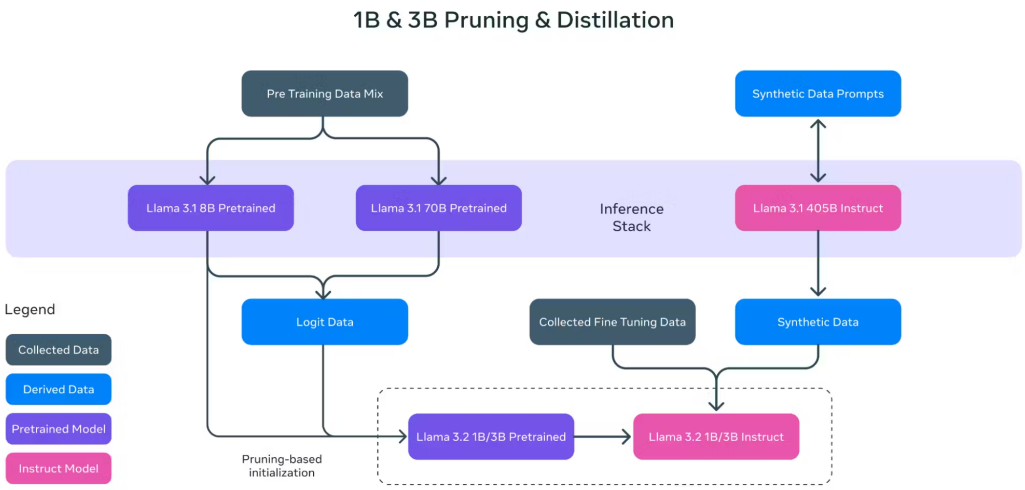

Figura 7: Arquitetura do LlaMA 3.2. Estrutura de treinamento utilizando destilação de conhecimento e pruning. (Fonte: Meta AI)

Entre as variantes da família, destaca-se o LlaMA 3.2-1B, que é a versão mais compacta com aproximadamente, 1 bilhão de parâmetros. Apesar de seu tamanho reduzido, essa versão demonstra um desempenho notável, resultado da utilização de técnicas como *pruning* e distilação de conhecimento. Essas técnicas permitem ao modelo manter uma alta performance enquanto reduz o consumo de recursos computacionais. O LlaMA 3.2-1B foi selecionado por seu equilíbrio ideal entre capacidade de processamento e eficiência, além de ser comparável a outros modelos menores usados nos experimentos.

2.4.3. Bormer

Inspirado no *PolyFormer*, o *Bormer* é uma arquitetura projetada como uma alternativa mais leve e rápida para problemas exclusivamente geométricos. Ao contrário do *PolyFormer*, que processa tanto imagens quanto comandos de texto, seus principais diferenciais residem no cálculo da função de perda e na forma como realiza o embedding dos pontos das sequências.

No núcleo do *Bormer* está uma arquitetura *Transformer* de texto tradicional, mas com a adição de duas *heads* acopladas à saída do *decoder*, responsáveis pela predição das coordenadas e da classe de cada *token* previsto, como apresentado na Figura 8.

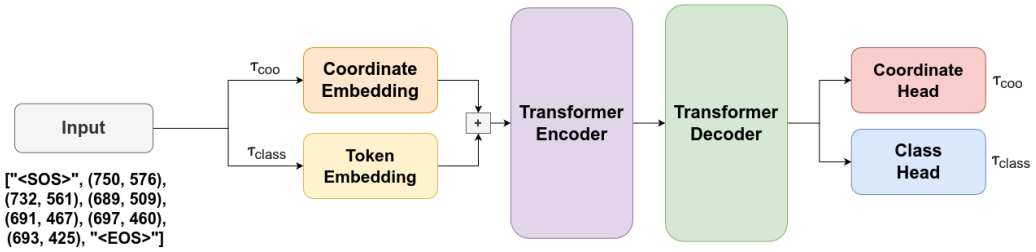

Figura 8: Arquitetura do *Bormer*.

Dessa forma, cada predição produz dois resultados: o próximo ponto e a classe do *token* associada a esse ponto. As classes de *tokens* existentes no *Bormer* incluem:

- ⟨SOS⟩: Marcador de início de sequência.
- ⟨EOS⟩: Marcador de fim de sequência.
- ⟨COO⟩: Marcador de coordenadas, indicando que o *token* contém um ponto geométrico.
- ⟨SEP⟩: Marcador de separação entre diferentes partes de uma figura.

Esse mecanismo motiva o uso de uma função de perda mista, composta pela soma entre a perda tradicional de entropia cruzada, calculada para todos os casos, e pela perda de regressão quadrática média (MSE), aplicada apenas para *tokens* de classe de coordenadas, como pode ser observado na Equação 1. A inclusão da perda regressiva na função objetiva oferece uma vantagem significativa em relação aos modelos de linguagem tradicionais, pois aumenta a penalidade para os erros mais acentuados enquanto reduz o impacto de erros próximos ao ponto correto. Esse ajuste proporciona uma redução substancial das alucinações do modelo.

$$\mathcal{L}_{Bormer} = \sum_{i \in \text{classes}} \mathcal{L}_{\text{CE}}(\tau^{\text{i}}_{class}, \hat{\tau}^{\text{i}}_{class}) + \sum_{j \in \text{coords}} \mathcal{L}_{\text{MSE}}(\tau^{\text{j}}_{coo}, \hat{\tau}^{\text{j}}_{coo}) \qquad (1)$$

[2]

- $\mathcal{L}_{Bormer}$: Função de perda total do modelo.

[2]Quando $\tau^{\text{i}}_{class} \neq \langle\text{COO}\rangle \Rightarrow \mathcal{L}_{\text{MSE}} = 0$

- $\mathcal{L}_{\text{CE}}(\tau^{\text{i}}_{class}, \hat{\tau}^{\text{i}}_{class})$: Perda de entropia cruzada, calculada para o token i, onde τ^{i}_{class} é a classe verdadeira e $\hat{\tau}^{\text{i}}_{class}$ é a probabilidade prevista da classe.

- $\mathcal{L}_{\text{MSE}}(\tau^{\text{j}}_{coo}, \hat{\tau}^{\text{j}}_{coo})$: Perda de erro quadrático médio, calculada para o token j, onde τ^{j}_{coo} é a coordenada verdadeira e $\hat{\tau}^{\text{j}}_{coo}$ é a coordenada prevista.

- classes: Conjunto de índices das classes dos tokens que estão sendo avaliados.

- coords: Conjunto de índices das coordenadas dos tokens que estão sendo avaliados.

Outro desafio nos modelos de linguagem convencionais está relacionado ao tokenizador, que geralmente é projetado para operar com uma linguagem completa, mas acaba subutilizado em problemas puramente numéricos. Em vocabulários convencionais, os dígitos são tratados de forma independente, o que aumenta o número de tokens necessários para o *embedding* de uma única coordenada. Esse fator contribui para uma eficiência reduzida e maior complexidade na representação de dados numéricos.

Para mitigar esse problema, foi desenvolvido um *coordinate embedding* especial para o *Bormer*, com o objetivo de reduzir o número de tokens e preservar melhor as propriedades das coordenadas. Esse *embedding* é baseado em um *TokenEmbedding* tradicional, mas utiliza um vocabulário restrito, composto apenas por dígitos em um intervalo definido (por padrão, 0–2000). Essa técnica é aplicada às coordenadas X e Y de cada token, cujos *embeddings* são concatenados e somados ao *embedding* da classe do *token* associado, como ilustrado na Figura 9.

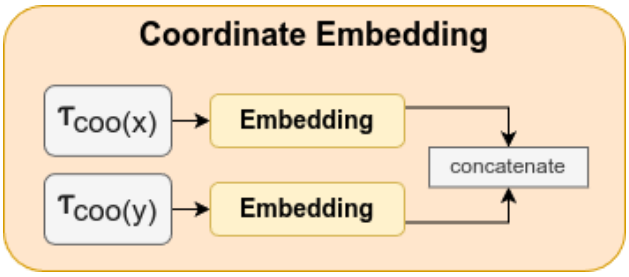

Figura 9: Arquitetura do *Coordinate Embedding*.

Ao contrário de abordagens anteriores, como no *PolyFormer*, que utiliza um vocabulário bidimensional para representar as coordenadas, o *Bormer* adota uma estrutura unidimensional com a finalidade de otimizar o uso da memória e desempenho, fundamentais para sua proposta de leveza e rapidez.

A versão do *Bormer* utilizada no experimento, contém aproximadamente 6 milhões de parâmetros, sendo o modelo mais rápido e leve entre os avaliados.

2.5. Protocolo de Experimento

Os experimentos de treinamento envolveram modelos de diferentes escalas, com uma variação de 6 milhões a 1 bilhão de parâmetros. Devido a essa grande discrepância em número de parâmetros, métricas como o número de épocas ou iterações não refletem adequadamente as diferenças significativas no tempo de treinamento entre os modelos. Para assegurar uma comparação justa, estabeleceu-se uma métrica uniforme de tempo de treinamento: cada arquitetura foi treinada durante um período fixo de 24 horas. Esse critério permitiu um número adequado de iterações para todas as arquiteturas testadas, e garantiu uma análise comparativa mais equilibrada.

De forma semelhante aos treinamentos realizados para modelos de linguagem tradicionais, foi inicialmente realizado um pré-treinamento em todas as arquiteturas. Esse pré-treinamento consistiu em fornecer a imagem da folha completa ao modelo, com o objetivo de replicá-la, como ilustrado na Figura 10. Esse processo inicial é essencial para preparar os modelos para tarefas subsequentes, ao aprimorar a capacidade de reconhecimento e reprodução da estrutura da folha.

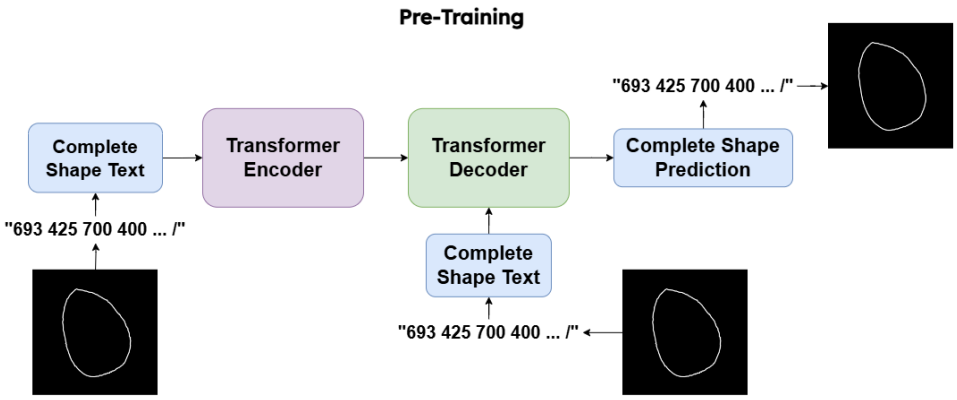

Figura 10: Procedimento de pré-treinamento.

Após o pré-treinamento, iniciaram-se os treinamentos efetivos das redes. Nessa fase, os modelos recebiam imagens de folhas degradadas e tinham a tarefa de prever a versão completa dessas folhas, conforme pode se observar na Figura 11. Esses treinamentos visavam habilitar os modelos a reconstruir áreas danificadas, e permitir uma análise detalhada de sua capacidade de generalização e precisão na recomposição das folhas.

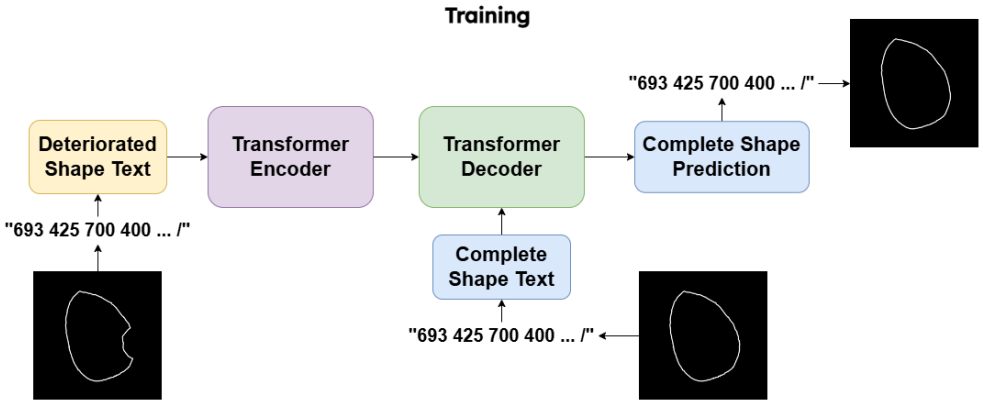

Figura 11: Procedimento de treinamento.

O modelo LlaMA 3.2-1B foi treinado com a utilização do framework Unsloth, uma plataforma que viabiliza o treinamento de grandes modelos de linguagem em GPUs comerciais tradicionais (Unsloth, 2024). Esse framework permite a aplicação de técnicas avançadas, como *gradient checkpointing*, *LoRA* e quantização, que foram utilizadas neste estudo para otimizar a eficiência e reduzir o uso de memória durante o treinamento. Utilizou-se uma versão pré-treinada do LlaMA, disponível dentro do próprio framework, como ponto de partida para os experimentos.

Os hiperparâmetros configurados para o treinamento deste modelo foram: o tamanho de lote (*batch size*) ajustado para 8; a taxa de aprendizado (*learning rate*) de 0.00001; o otimizador AdamW; e a quantização de 8 bits. A quantização foi aplicada devido à grande quantidade de parâmetros do modelo, a fim de otimizar o uso de memória e possibilitar seu treinamento.

O modelo T5-Small foi treinado integralmente com o uso do framework Hugging Face, que oferece suporte ao treinamento de modelos de linguagem em múltiplas arquiteturas (Face, 2024a). Para este experimento, utilizou-se uma versão pré-treinada do T5-Small disponibilizada pela Google em seu re-

positório no Hugging Face (Face, 2024b). Dada a arquitetura relativamente compacta do modelo T5-Small, foi realizado um treinamento *from scratch* com o objetivo de avaliar o impacto do pré-treinamento em tarefas de linguagem natural sobre a capacidade de generalização do modelo para problemas de natureza geométrica. Essa abordagem nos permitiu investigar se o conhecimento prévio adquirido durante o pré-treinamento é essencial para o desempenho em tarefas que exigem um tipo diferente de raciocínio.

Os principais hiperparâmetros utilizados foram configurados da seguinte forma: o tamanho de lote (*batch size*) ajustado para 128; o comprimento máximo de sequência (*max length*) definido em 500; a taxa de aprendizado (*learning rate*) de $0,0001$; e o otimizador AdamW.

O *Bormer* foi desenvolvido e treinado com o uso exclusivo do PyTorch[3] como framework base. Os treinamentos foram realizados com inicialização aleatória dos pesos em todas as camadas. Diversas configurações de tamanho do *Bormer* foram testadas, incluindo variações no tamanho do *embedding*, número de *attention heads*, e quantidades de camadas nos blocos de *encoder* e *decoder*.

Entretanto, a configuração principal utilizada nos experimentos foi a versão padrão (*default*) do *Bormer*, que possui 4 camadas de *encoder*, 4 camadas de *decoder*, 4 *attention heads* por célula, e tamanho do *embedding* fixado em 256.

Os hiperparâmetros do treinamento foram definidos como: tamanho do *batch* (*batch size*) de 128, comprimento máximo da sequência (*max length*) de 500, taxa de aprendizado (*learning rate*) de $0,0001$, e o otimizador Adam.

Os experimentos foram realizados em um computador com CPU AMD Ryzen 9 5900x @ 3,70 GHz x 12, 32 GB de memória e uma NVIDIA GeForce RTX 3090 (10496 núcleos de Arquitetura de Dispositivo Unificado de Computação - CUDA e 24 GB de memória de vídeo) e sistema operacional Ubuntu 22.04.

A validação de modelos de deep learning é uma importante etapa para garantir que o desempenho observado durante o treinamento se generalize bem para novos dados. Para isso, métricas de validação são usadas para avaliar a eficácia do modelo ao longo do tempo, e permitir ajustes finos que maximizam a sua precisão e confiabilidade, como a *Mean Absolute Error*(MAE) (Willmott and Matsuura, 2005) e o coeficiente de determinação,

[3]PyTorch

comumente referido como R^2 (Peirce et al., 2021). O Erro Absoluto Médio (MAE) e o coeficiente de determinção podem ser definidos conforme as equações 2 e 3, respectivamente.

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|d_i - \hat{d}_i| \tag{2}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(d_i - \hat{d}_i)^2}{\sum_{i=1}^{N}(d_i - \bar{d})^2} \tag{3}$$

Em que, d_i representa o percentual de desfolha real para o i-ésimo exemplo, $\hat{d}_i$ é o percentual de desfolha predito pelo modelo para o i-ésimo caso, $\bar{d}$ é a média dos valores reais de desfolha e N é o número de exemplos no conjunto de validação.

A métrica MAE, reflete a média das diferenças absolutas entre os valores reais e preditos, e oferece uma interpretação direta da magnitude dos erros (Willmott and Matsuura, 2005). Essa métrica é de particular importância porque, ao medir diretamente o erro absoluto médio, ela nos fornece uma indicação clara de quão próximos os valores preditos estão dos valores reais em termos de desfolha. Quanto menor o valor de MAE, mais precisa é a predição, o que indica que o modelo consegue captar com maior precisão a geometria das folhas e inferir corretamente as áreas faltantes ou degradadas.

Já a métrica R^2 é especialmente útil para entender o quão bem o modelo captura a relação entre as variáveis de entrada e saída. Uma das maneiras mais comuns de visualizar os resultados dessa métrica é por meio de gráficos que indicam o desempenho do modelo de regressão, como, por exemplo, um gráfico de dispersão entre os valores reais e os valores preditos.

Idealmente, se o modelo estiver com um bom desempenho, os pontos no gráfico estarão concentrados ao longo de uma linha reta com inclinação 1, que representa a correspondência exata entre d e $\hat{d}$. Porém, quando os pontos de dispersão se afastam dessa linha de correspondência perfeita, o valor de R^2 diminui, o que indica uma discrepância maior entre os percentuais de desfolha reais e preditos.

A interpretação gráfica é útil para identificar padrões, como possíveis regiões do espaço de entrada onde o modelo tem um desempenho pior ou melhor, como folhas com desfolhas muito altas ou muito baixas.

Além disso, para tarefas de segmentação, a métrica de *Mean Intersection over Union (mIoU)* é amplamente utilizada para avaliar o desempenho em

modelos que precisam identificar e demarcar áreas específicas de uma imagem, como regiões danificadas de folhas. A $mIoU$ mede a sobreposição entre as áreas previstas e as áreas reais das classes de interesse, sendo definida conforme a equação 4.

$$\text{mIoU} = \frac{1}{C} \sum_{c=1}^{C} \frac{|A_c \cap B_c|}{|A_c \cup B_c|} \tag{4}$$

Em que, C é o número total de classes, A_c representa a área prevista para a classe c, e B_c é a área real correspondente a essa mesma classe. O termo *Intersection over Union (IoU)* calcula a proporção entre a interseção e a união das áreas prevista e real, de modo que quanto maior o valor, melhor o desempenho do modelo em delinear com precisão as regiões específicas.

A média da IoU entre todas as classes, ou $mIoU$, é uma métrica essencial em segmentação, pois fornece uma medida quantitativa de similaridade entre a segmentação prevista e a real, facilitando a avaliação do modelo em contextos onde a precisão espacial é crucial.

3. Resultados & Discussões

3.1. Resultados Quantitativos

Tabela 1: Comparação de Modelos com Métricas de Desempenho

Model	**MAE**	**R^2**	**mIoU**
T5-Small-Pre-Trained	18.35	-3.76	80.4
T5-Small-From-Scratch	13.34	-5.15	72.59
LlaMA 3.2-1B	41.43	-23.22	32.15
Bormer	**3.92**	**0.68**	**87.06**

Na Tabela 1 foram apresentados as comparações quantitativas entre os modelos *T5-Small-Pre-Trained*, *T5-Small-From-Scratch*, *LlaMA 3.2-1B* e *Bormer* em relação as métricas MAE, R^2 e mIoU para avaliar a eficácia na predição do percentual de desfolha em folhas de soja, que foram descritas na subseção 2.5 da Metodologia.

A análise dos resultados mostrou que o modelo *Bormer* apresentou o melhor desempenho geral nas três métricas avaliadas, com o menor MAE

(3.92), o maior valor de R^2 (0.68) e o maior mIoU (87.06). Em termos de MAE, o *Bormer* mostrou uma diferença significativa em relação aos demais modelos, o que indica uma maior precisão na estimativa do percentual de desfolha.

Ao comparar o modelo *T5-Small-Pre-Trained* com o *T5-Small-From-Scratch*, observou-se uma redução de aproximadamente 27.3% no MAE, o que sugere que o modelo treinado do zero (*From-Scratch*) pode ter se adaptado melhor ao contexto específico de desfolha, embora ainda apresente uma margem de erro considerável em relação ao *Bormer*. O modelo *LlaMA 3.2-1*B, por outro lado, apresentou um MAE muito mais elevado (41.43), o que representa uma precisão bem menor na estimativa de desfolha.

Para a métrica R^2, o *Bormer* é o único modelo com um valor positivo (0.68), o que indica que ele conseguiu capturar a variabilidade dos dados de desfolha de maneira mais consistente. Os demais modelos, especialmente o *LlaMA 3.2-1B*, apresentaram valores negativos de R^2, o que sugere uma baixa capacidade de explicação da variância nos dados de desfolha.

Em relação ao mIoU, o *Bormer* novamente se destacou com a maior pontuação (87.06) e isso representa uma melhoria de aproximadamente, 8.3% em relação ao *T5-Small-Pre-Trained* e de 19.8% em relação ao *T5-Small-From-Scratch*. O *LlaMA 3.2-1B*, no entanto, teve um desempenho significativamente inferior, com mIoU de 32.15, que evidencia uma baixa capacidade de segmentação precisa das áreas de desfolha.

3.2. Resultados Qualitativos

Para complementar a análise quantitativa, também foi realizado uma análise qualitativa dos modelos utilizados para a reconstrução de folhas de soja desfolhadas. Na Figura 12, são apresentados os resultados visuais de cada modelo — *LlaMA 3.2-1B*, *T5-Small Pre Trained*, *T5-Small From Scratch* e *Bormer* — comparados com a folha original e a folha alvo.

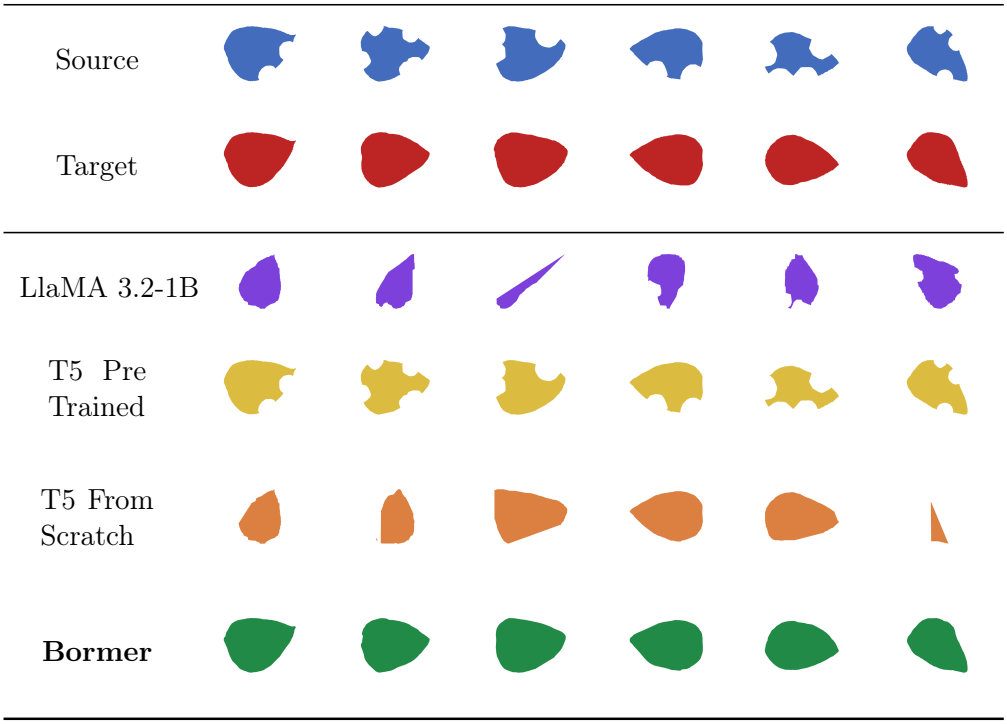

Figura 12: Resultados de diferentes modelos para exemplos em comum.

Assim, conforme a Figura 12, os aspectos qualitativos mais relevantes de cada modelo foram:

- O modelo *LlaMA 3.2-1B* apresentou uma baixa capacidade de reconstrução das áreas desfolhadas, com reconstruções fragmentadas e formas desconexas. Observou-se que algumas folhas completadas pelo modelo possuem formatos que não correspondem à estrutura original, o que indica que o modelo não capturou adequadamente as características estruturais das folhas. Esse comportamento reflete um subajuste do modelo, e resulta em uma reconstrução pouco realista e inconsistente.
- O modelo *T5-Small Pre Trained* apresentou limitações significativas na tarefa de reconstrução das folhas desfolhadas. Observou-se que o modelo basicamente, reproduziu as formas incompletas fornecidas na entrada, sem adicionar as informações necessárias para completar as áreas desfolhadas. Esse comportamento indica que o modelo não conseguiu extrair de forma eficaz as características visuais e estruturais das

folhas, ou seja, mostrou-se incapaz de realizar a tarefa de completude das folhas de maneira satisfatória. Esse comportamente indica uma forte influência das tarefas de linguagem na qual modelos como o T5 foram pré treinados, onde é comum que o modelo repita em sua saída uma grande parte de sua entrada.

- O modelo *T5-Small From Scratch* apresentou uma leve melhoria em relação ao modelo anterior. As áreas desfolhadas reconstruídas foram mais uniformes, e o formato das folhas reconstruídas foi mais próximo do ideal. Embora ainda não seja perfeitamente preciso, esse modelo mostrou uma capacidade aprimorada de captar a estrutura geral das folhas, o que resultou em uma reconstrução qualitativamente superior em relação ao *T5 Pre Trained*.

- O modelo *Bormer* se destacou em relação aos demais, ao apresentar reconstruções que se assemelham bastante ao alvo ideal, tanto em forma quanto em completude das áreas desfolhadas. As folhas reconstruídas pelo *Bormer* apresentaram contornos suavizados e preservaram a integridade visual das folhas originais. Esse modelo demonstrou uma capacidade robusta de generalizar a estrutura das folhas, ao completar as áreas desfolhadas com alta precisão. Assim, o *Bormer* foi o mais eficaz para a tarefa de reconstrução de folhas desfolhadas.

Logo, o *Bormer* mostrou o melhor desempenho qualitativo, sendo capaz de reconstruir as folhas de maneira precisa e completa, uma característica essencial para aplicações que exigem alta fidelidade visual. Por outro lado, o *LlaMA 3.2-1B* apresentou os piores resultados, com reconstruções desconexas e inconsistentes. Os modelos *T5-Small* demonstraram desempenho intermediário, sendo o *T5-Small From Scratch* ligeiramente superior ao *T5 Pre Trained*.

A Figura 13 apresenta os resultados das predições dos percentuais de desfolha obtidos pelo modelo *Bormer* em comparação com os percentuais de desfolha reais. Observou-se que o modelo *Bormer* demonstrou maior consistência em situações onde a desfolha estava presente em algum nível menos elevado, indicando sua capacidade de capturar padrões relacionados à perda de área foliar. No entanto, é possível perceber que o modelo tende a superestimar o percentual de desfolha em casos onde esse valor ultrapassa a faixa de 20%.

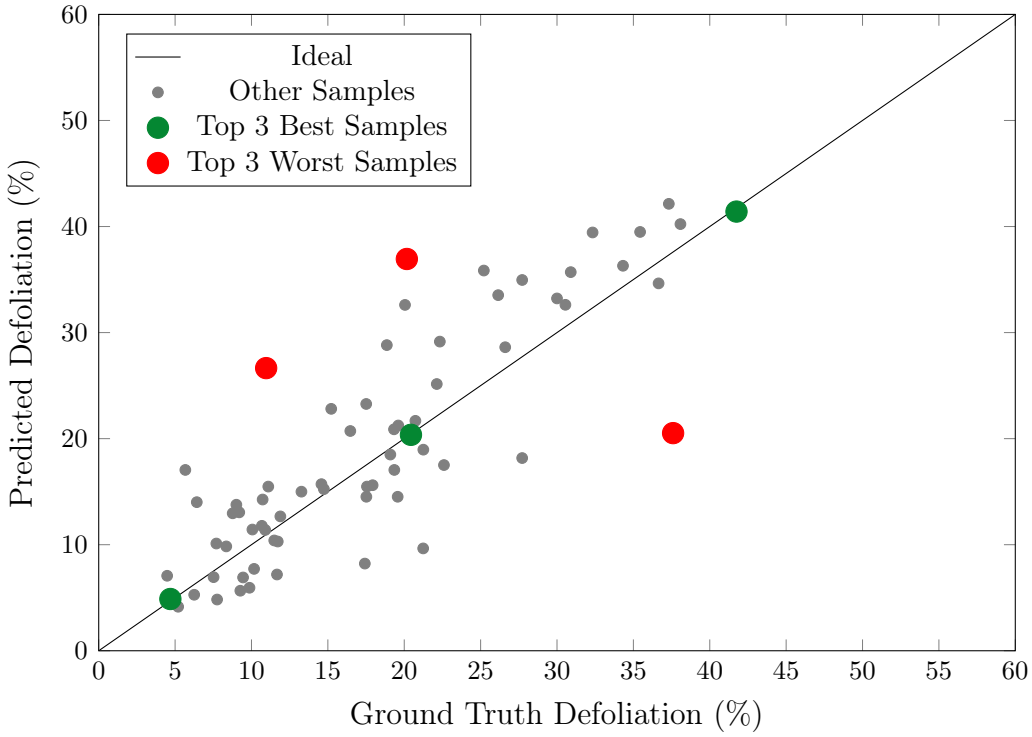

Figura 13: Gráfico do Coeficiente de Determinação (R^2) do modelo *Bormer*

A Figura 14 apresenta visualmente os três melhores e três piores resultados do modelo, identificados na Figura 13 pelos pontos em verde e vermelho, respectivamente. Observa-se que, nos piores casos, o modelo apresentou um certo grau de "alucinação", uma vez que as predições não apenas divergem dos valores-alvo, mas também mostram mudanças significativas em relação à entrada original. Esse comportamento sugere que o modelo pode ter dificuldade em manter a consistência das características das folhas em situações específicas, levando a previsões que se distanciam consideravelmente da realidade esperada.

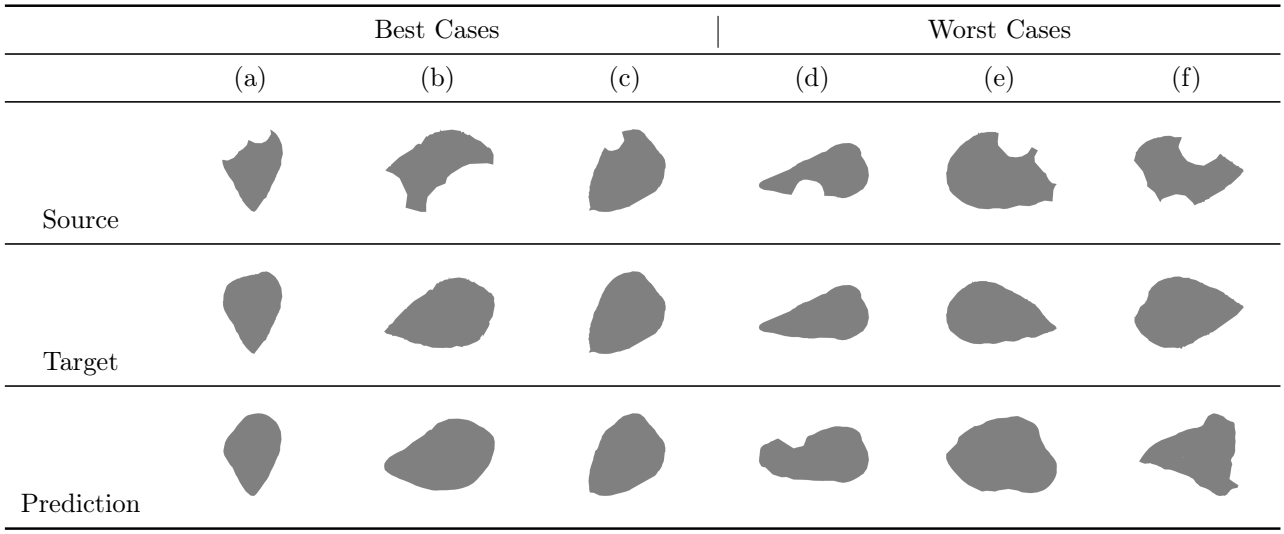

Figura 14: Comparativo de predições do modelo *Bormer*: Melhores e Piores Casos

3.3. Modelos de Linguagem Pré-Treinados na Predição Geométrica

Modelos amplamente utilizados em diversos tipos de tarefas de linguagem, como o *T5-Small*, demonstram grande eficiência em múltiplos domínios quando adaptados por *fine-tuning*. Contudo, ao serem aplicados à predição de coordenadas geométricas, os resultados indicaram que versões treinadas do zero superam significativamente as pré-treinadas.

Essa situação ocorre, pois a tarefa de predição de coordenadas geométricas é fundamentalmente distinta da predição de palavras. Em contextos de linguagem, as correlações entre *embeddings* numéricos geralmente estão relacionadas a datas, cálculos ou quantidades específicas, o que limita a aplicabilidade direta desses pesos para a tarefa de previsão de coordenadas geométricas, que se assimila muito mais como um problema de regressão.

3.4. Desempenho

Durante as fases de treinamento e inferência, foi observada uma clara vantagem do *Bormer* em relação à velocidade de inferência e ao uso de memória. Essa vantagem pode ser atribuída ao número significativamente menor de parâmetros, como ilustrado na Tabela 2, que apresenta a comparação do tempo nescessário para realizar a inferência do dataset de teste inteiro. Além disso, uma peculiaridade do *Bormer* é o tamanho do contexto que ele consegue processar. Ao contrário dos modelos de linguagem tradicionais, que nem sempre

tratam os valores numéricos como *embeddings* únicos, podendo gerar múltiplos tokens para uma única coordenada, o *Bormer* gera um embedding único para cada ponto (X, Y). Isso permite, no mínimo, dobrar a capacidade do contexto, utilizando a mesma quantidade de memória que os tokenizadores tradicionais em modelos de linguagem.

Tabela 2: Tempo de Inferência entre os modelos analisados

Modelo	**Tempo de Inferência no Dataset de Teste [s]**
T5-Small	217
LlaMA 3.2-1B	2621
Bormer	**18**

3.5. Alucinações

As alucinações são um problema recorrente em modelos generativos, frequentemente resultando em predições que não correspondem à realidade do contexto analisado, como valores de desfolha em posições impossíveis ou irrealistas. Ao observar os resultados, é evidente que todos os modelos analisados apresentam algum grau de alucinação. No entanto, o *Bormer* se distingue por gerar alucinações com uma natureza geométrica mais controlada. Em vez de extrapolações errôneas, como a geração de pontos totalmente desconexos da forma original, as alucinações do *Bormer* tendem a gerar formas mais coerentes, que não comprometem tanto a integridade geométrica da previsão. Essa resistência a alucinações mais drásticas foi um dos objetivos centrais da inclusão de uma função de perda de regressão no modelo. Em comparação, os modelos de linguagem tradicionais comparados frequentemente geram formas que variam de maneira extrema, como pode ser observado em 12, com formatos irregulares e desproporcionais.

4. Conclusão

O Bormer se destacou como uma solução eficiente e precisa para reconstrução de folhas de soja, superando modelos que compõem o estado da arte nesse tipo de tarefa. Com uma abordagem centrada em formas, o modelo demonstrou capacidade superior em prever completar as áreas degradadas, combinando alta precisão geométrica com baixo consumo de recursos computacionais. A introdução de um coordinate embedding otimizado e uma

função de perda mista se demonstraram cruciais para redução de alucinações e aumento da robustez das predições. Além de contribuir significativamente para a agricultura de precisão, o Bormer apresenta potencial para aplicações em outros domínios que demandam restauração geométrica confiável, reforçando sua versatilidade e relevância.

Durante a preparação deste trabalho, os autores utilizaram o ChatGPT com o objetivo de realizar revisões quanto à ortografia, melhorar a coerência e coesão do texto. Após o uso desta ferramenta, os autores revisaram e editaram o conteúdo conforme necessário e assumem plena responsabilidade pelo conteúdo da publicação.

5. Referências

Aloun, M.S., Hitam, M.S., Yussof, W.N.J.H.W., Bachok, Z., 2022. A review paper on image segmentation techniques based on colour and texture features. Nucleation and Atmospheric Aerosols doi:10.1063/5.0114074.

Bressan, P.O., Junior, J.M., Correa Martins, J.A., de Melo, M.J., Gonçalves, D.N., Freitas, D.M., Marques Ramos, A.P., Garcia Furuya, M.T., Osco, L.P., de Andrade Silva, J., Luo, Z., Garcia, R.C., Ma, L., Li, J., Gonçalves, W.N., 2022. Semantic segmentation with labeling uncertainty and class imbalance applied to vegetation mapping. International Journal of Applied Earth Observation and Geoinformation 108, 102690. URL: https://www.sciencedirect.com/science/article/pii/S0303243422000162, doi:https://doi.org/10.1016/j.jag.2022.102690.

Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2017. Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 .

Chen, T., Saxena, S., Li, L., Fleet, D.J., Hinton, G., 2022. Pix2seq: A language modeling framework for object detection. URL: https://arxiv.org/abs/2109.10852, arXiv:2109.10852.

da Silva, L.A., Bressan, P.O., Gonçalves, D.N., Freitas, D.M., Machado, B.B., Gonçalves, W.N., 2019. Estimating soybean leaf defoliation using convolutional neural networks and synthetic images. Computers and Electronics in Agriculture 156, 360–368. URL: https://www.sciencedirect.

com/science/article/pii/S0168169918307907, doi:https://doi.org/10.1016/j.compag.2018.11.040.

Douglas, D.H., Peucker, T.K., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: The International Journal for Geographic Information and Geovisualization 10, 112–122.

Face, H., 2024a. Hugging face framework. https://huggingface.co/.

Face, H., 2024b. T5: Text-to-text transfer transformer. https://huggingface.co/google-t5/t5-small.

Fracarolli, J.A., Pavarin, F.F.A., Castro, W., Blasco, J., 2020. Computer vision applied to food and agricultural products. Revista Ciência Agronômica 51, e20207749. URL: https://doi.org/10.5935/1806-6690.20200087, doi:10.5935/1806-6690.20200087.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, pp. 2961–2969.

Hughes, D.P., Salathe, M., 2015. An open access repository of images on plant health to enable the development of mobile disease diagnostics. URL: https://arxiv.org/abs/1511.08060, arXiv:1511.08060.

Jeronymo, C.E., 2023. Aplicação do algoritmo de douglas-peucker na redução de dados hidro-telemétricos, in: XXV Simpósio Brasileiro de Recursos Hídricos, Associação Brasileira de Recursos Hídricos, Sergipe, Brasil. URL: https://anais.abrhidro.org.br/job.php?Job=14987. iSSN 2318-0358.

Ji, X., Lucas, T., Froehlich, B., 2024. Trapped in texture bias? a large scale comparison of deep instance segmentation. arXiv preprint arXiv:2401.09109 URL: https://ar5iv.org/abs/2401.09109.

Kamilaris, A., Prenafeta-Boldú, F.X., 2018. Deep learning in agriculture: A survey. Computers and Electronics in Agriculture 147, 70–90. URL: https://www.sciencedirect.com/science/article/pii/S0168169917308803, doi:https://doi.org/10.1016/j.compag.2018.02.016.

Liu, J., Ding, H., Cai, Z., Zhang, Y., Satzoda, R.K., Mahadevan, V., Manmatha, R., 2023. Polyformer: Referring image segmentation as sequential polygon generation. URL: https://arxiv.org/abs/2302.07387, arXiv:2302.07387.

Peirce, C., et al., 2021. The coefficient of determination r-squared is more informative than other metrics for regression analysis. PeerJ Computer Science 7, e623. doi:10.7717/peerj-cs.623.

Purkait, P., Zach, C., Reid, I., 2019. Seeing behind things: Extending semantic segmentation to occluded regions. URL: https://arxiv.org/abs/1906.02885, arXiv:1906.02885.

Raffel, C., Shazeer, N., Roberts, A., Lee, P.J., Narang, S., Matena, M., Zhou, Y., Fiedler, L., Li, N., Liu, P., et al., 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. The Journal of Machine Learning Research 21, 1–67.

Rinaldi, A.M., Russo, C., Tommasino, C., 2023. Automatic image captioning combining natural language processing and deep neural networks. Results in Engineering 18, 101107. URL: https://www.sciencedirect.com/science/article/pii/S2590123023002347, doi:https://doi.org/10.1016/j.rineng.2023.101107.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G., 2023. Llama: Open and efficient foundation language models. URL: https://arxiv.org/abs/2302.13971, arXiv:2302.13971.

Unsloth, E., 2024. Unsloth: Framework for training large models on commercial gpus. https://github.com/unslothai/unsloth.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. CoRR abs/1706.03762. URL: http://arxiv.org/abs/1706.03762, arXiv:1706.03762.

Willmott, C.J., Matsuura, K., 2005. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. Climate research 30, 79–82.

Yu, Y., Wang, C., 2023. Techniques and challenges of image segmentation: A review. Electronics 12. doi:10.3390/electronics12051199.

Yuan, X., Kortylewski, A., Sun, Y., Yuille, A., 2021. Robust instance segmentation through reasoning about multi-object occlusion. URL: https://arxiv.org/abs/2012.02107, arXiv:2012.02107.