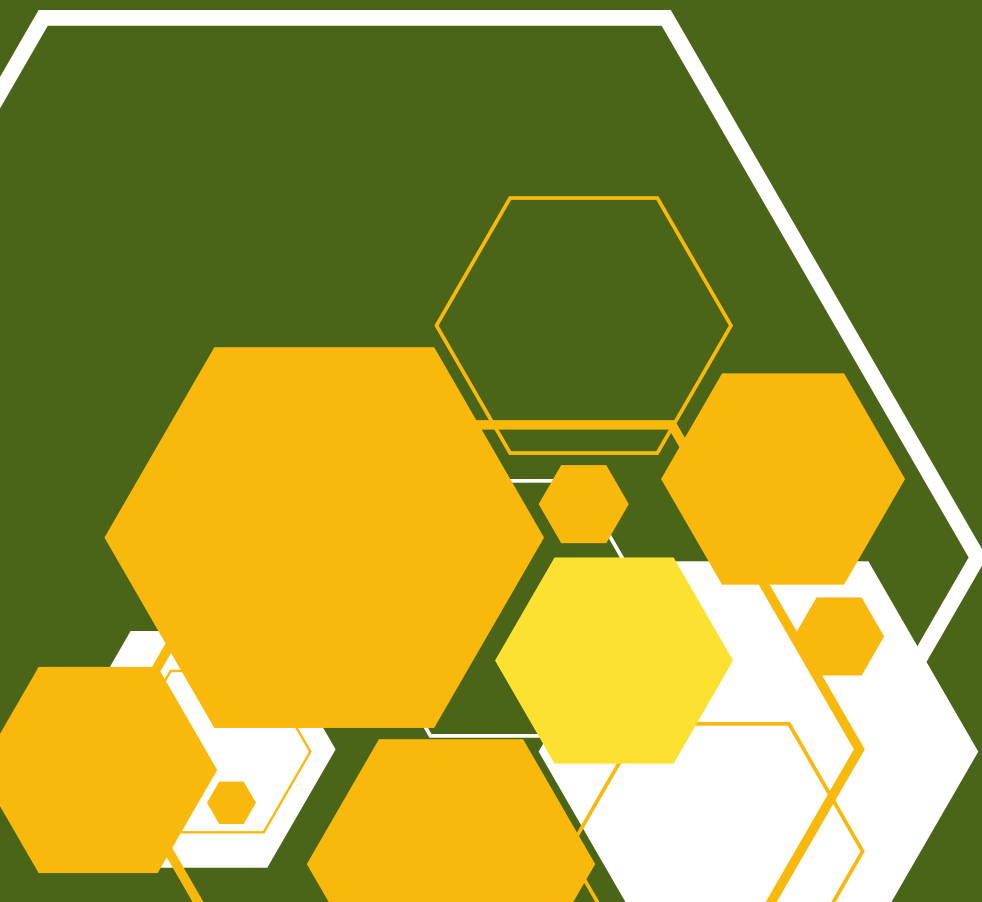
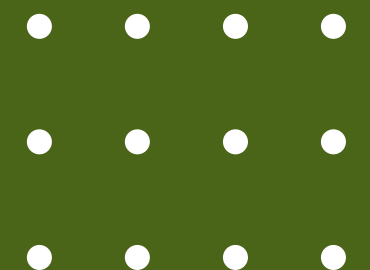


Etapa 2

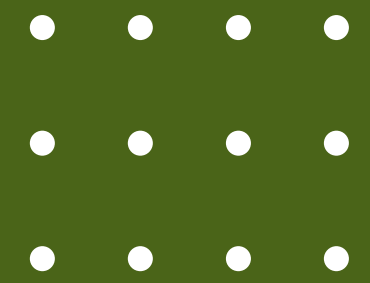
Análise Crítica do Related Notebook - CNN vs Transfer Learning

**Dataset: Agricultural crops image classification,
829 imagens, 30 classes**

Alunos: Alexia Rocha
Darc Mary
Denise Ramos



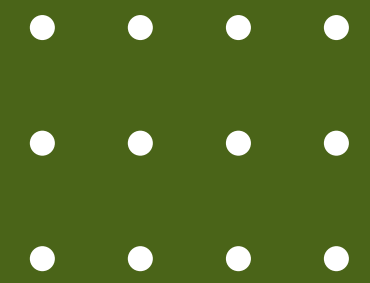
Entendendo os modelos comparados CNN



- Arquitetura construída manualmente
- Aprende padrões diretamente a partir das imagens
- Requer mais dados para evitar overfitting
- Modelo do estudo: CNN com 85M de parâmetros



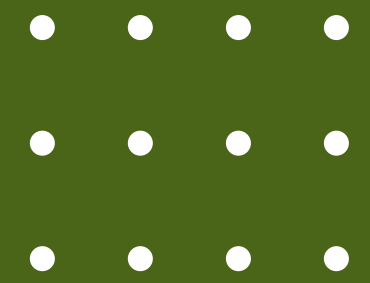
Estrutura da CNN usada no notebook



- 3 blocos com Conv2D (32, 64, 128) + MaxPooling2D
- Flatten() → Dense(1024) com ReLU → Dropout(0.5) → Dense(30) com Softmax
- Otimizador: Adam
- Função de perda: categorical_crossentropy
- Batch size: 32
- Input shape: (224, 224, 3)
- Epochs: 50



Transfer Learning (VGG16)



- Utiliza rede pré-treinada em grande base de dados (ImageNet)
- Aproveita conhecimento já aprendido
- Ajuste final (fine-tuning) para o novo conjunto
- Mais eficiente com datasets pequenos

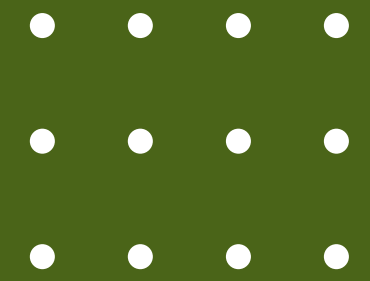


Estrutura do modelo com Transfer Learning

- Base: MobileNetV2(weights='imagenet', include_top=False)
- Input shape: (224, 224, 3)
- Camada de pooling: GlobalAveragePooling2D()
- Camada densa intermediária: Dense(128) com ReLU Dropout(0.5)
- Saída: Dense(28) com Softmax
- Otimizador: Adam
- Função de perda: categorical_crossentropy
- Batch size: 32
- Epochs: 50



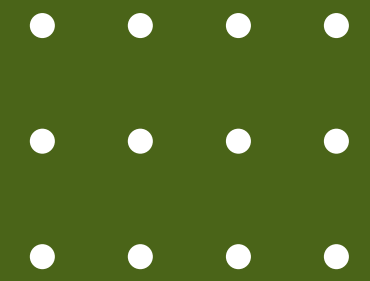
Boas práticas observadas



- Uso de Data Augmentation
- Comparação: CNN simples vs VGG16 (Transfer Learning)
- Aplicação de Dropout contra overfitting
- ImageDataGenerator com transformações variadas



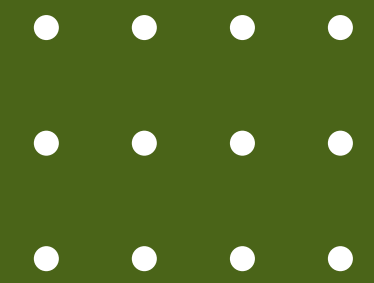
Limitações encontradas



- CNN com 85M de parâmetros
- Divisão dos dados sem estratificação
- Inconsistência nos tamanhos das imagens (150x150 vs 250x250)
- Uso de `os.makedirs()` sem `exist_ok=True`



Propostas de melhoria (1/2)



1. Padronizar tamanhos de imagem

- Usar o mesmo target_size em todas as etapas

2. Reduzir complexidade do modelo

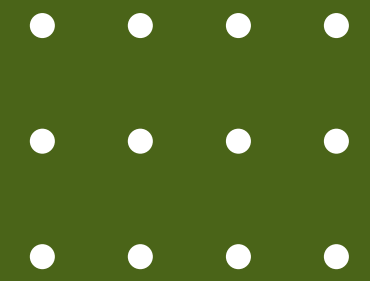
- Menos filtros/neuronios
- GlobalAveragePooling2D() no lugar de Flatten()
- Incluir BatchNormalization

3. Melhorar divisão dos dados

- Usar train_test_split com stratify



Propostas de melhoria (2/2)



4. Tratar duplicatas

- Remover ou manter pares no mesmo split

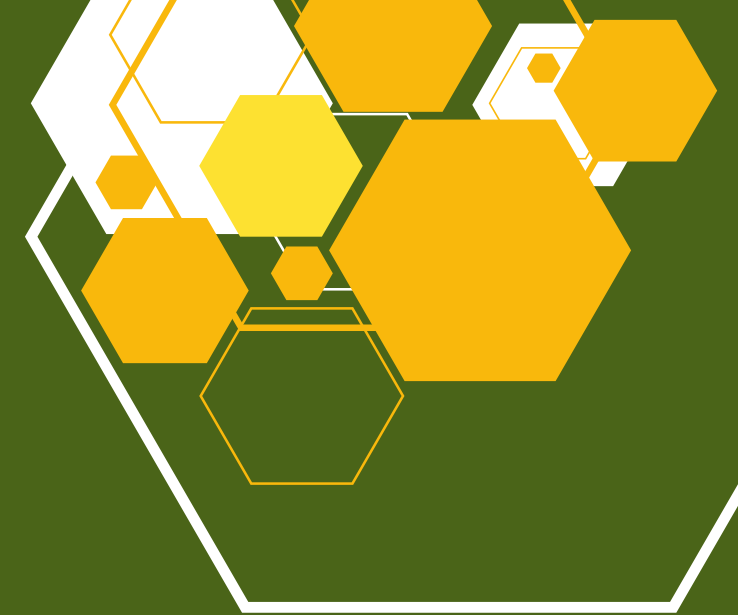
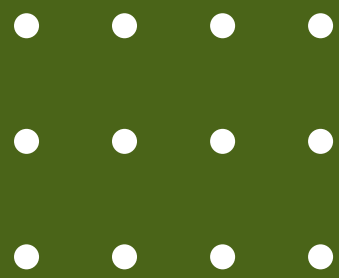
5. Testar validação cruzada simples

- Permite avaliar a estabilidade dos resultados em diferentes divisões.

6. Usar `class_weight` para balancear classes

- Ajuda o modelo a lidar melhor com classes minoritárias.





Etapa 2

Obrigada!

