

Classificação de Imagens de Satélite com o Dataset EuroSAT

Uma abordagem usando Redes Neurais Convolucionais

Luiz Felipe Elizeta dos S.

Machine Learning

12 de dezembro de 2025

Sumário

- 1 Introdução
- 2 O Dataset
- 3 Pré-processamento
- 4 Arquitetura do Modelo
- 5 Resultados
- 6 Conclusão

Objetivo:

- Classificar imagens de satélite em 10 categorias de uso e cobertura do solo.
- Utilizar o dataset EuroSAT (RGB).

Ferramentas Utilizadas:

- Python e Jupyter Notebook.
- TensorFlow e Keras para Deep Learning.
- TensorFlow Datasets ('tfds') para carregar os dados.
- Matplotlib para visualização.
- Scikit-learn para métricas de avaliação.

Carregamento e Divisão dos Dados

Os dados foram carregados diretamente via 'tensorflow datasets' com a seguinte divisão:

- **Treino:** 80% das imagens (21.600 imagens).
- **Validação:** 10% das imagens (2.700 imagens).
- **Teste:** 10% das imagens (2.700 imagens).

```
1 (ds_train, ds_val, ds_test), ds_info = tfds.load(  
2     'euosat/rgb',  
3     split=['train[:80%]', 'train[80%:90%]', 'train[90%:]'],  
4     as_supervised=True,  
5     with_info=True  
6 )  
7
```

O dataset contém 10 classes distintas de uso do solo:

- | | |
|------------------------|-----------------|
| 1 AnnualCrop | 6 Pasture |
| 2 Forest | 7 PermanentCrop |
| 3 HerbaceousVegetation | 8 Residential |
| 4 Highway | 9 River |
| 5 Industrial | 10 SeaLake |

Preparação dos Dados (Pipeline)

Para otimizar o treinamento, aplicamos transformações nos dados:

- **Resize:** Redimensionamento para 224×224 pixels (padrão ResNet).
- **Batching:** Agrupamento em lotes de 32 imagens.
- **Prefetching:** Carregamento assíncrono para não travar a GPU.
- **Data Augmentation:** Flip horizontal, rotação e ajuste de contraste (apenas no treino).

```
1 def preprocess(image, label):  
2     image = tf.image.resize(image, (224, 224))  
3     return image, label  
4  
5 train_ds = train_ds.map(preprocess).batch(32).prefetch(  
6     buffer_size=tf.data.AUTOTUNE)
```

Utilizamos a arquitetura **ResNet50** pré-treinada no ImageNet:

- **Base:** ResNet50 (sem o topo/camada final).
- **Pesos:** 'imagenet' (congelados).
- **Camadas Adicionais:**
 - Global Average Pooling.
 - Dense (256 neurônios, ReLU).
 - Batch Normalization e Dropout (0.3).
 - Camada de Saída (10 neurônios, Softmax).

Transfer Learning com ResNet50

```
1 base_model = tf.keras.applications.ResNet50(  
2     include_top=False, weights='imagenet', input_shape=(224,  
3     224, 3)  
4 )  
5 base_model.trainable = False # Congelar a base  
6  
7 inputs = tf.keras.Input(shape=(224, 224, 3))  
8 x = base_model(inputs, training=False)  
9 x = tf.keras.layers.GlobalAveragePooling2D()(x)  
10 x = tf.keras.layers.Dense(256, activation='relu')(x)  
11 outputs = tf.keras.layers.Dense(10, activation='softmax')(x)  
12 model = tf.keras.Model(inputs, outputs)
```


Configurações do treinamento:

- **Otimizador:** Adam (learning rate = 0.001).
- **Loss Function:** Sparse Categorical Crossentropy.
- **Épocas:** 15.
- **Callbacks:**
 - 'EarlyStopping': Para quando a 'val loss' não melhora.
 - 'ReduceLROnPlateau': Reduz o learning rate se estagnar.

O modelo alcançou uma performance robusta nos dados de teste:

Acurácia Global: 94.52%

Loss Global: 0.1702

Relatório de Classificação (Exemplo)

Métricas detalhadas por classe (Precision, Recall, F1-Score):

Classe	Precision	Recall	F1-Score	Support
AnnualCrop	0.93	0.97	0.95	276
Forest	0.97	0.97	0.97	286
HerbaceousVegetation	0.91	0.95	0.93	317
SeaLake	0.86	0.93	0.90	259
...
Weighted Avg	0.94	0.94	0.93	2700

- O uso de Transfer Learning com ResNet50 mostrou-se muito eficaz para o dataset EuroSAT, atingindo acurácia superior a 93% em poucas épocas.
- As técnicas de Data Augmentation e o uso de callbacks ajudaram a evitar overfitting e melhorar a generalização.
- A matriz de confusão (gerada no notebook) permite identificar quais classes o modelo confunde mais (ex: AnnualCrop vs. PermanentCrop).
- Gerou bons resultados com um ótimo tempo de execução.

Obrigado!