TensorFlow

TensorFlow Everywhere
Supervised Contrastive Learning

Dimitre Oliveira

Specialist Data Scientist @ Virtus, UFCG

Google Developer Expert on Machine Learning

/dimitre-oliveira-7a1a0113a

Keras

keras.io

About Keras
Getting started
Developer guides
Keras API reference
Code examples
Computer Vision
Natural language processing
Structured Data
Timeseries
Audio Data
Generative Deep Learning
Reinforcement learning
Quick Keras recipes
Why choose Keras?
Community & governance
Contributing to Keras

Code examples

Our code examples are short (less than 300 lines of code), focused demonstrations of vertical deep learning workflows.

All of our examples are written as Jupyter notebooks and can be run in one click in Google Colab, a hosted notebook environment that requires no setup and runs in the cloud. Google Colab includes GPU and TPU runtimes.

Code examples
Computer Vision
Natural language processing
Structured Data
Timeseries
Audio Data
Generative Deep Learning
Reinforcement learning
Quick Keras recipes
▻ Adding a new code example

Computer Vision

- Image classification from scratch
- Simple MNIST convnet
- Image segmentation with a U-Net-like architecture
- 3D Image Classification from CT Scans
- OCR model for reading Captchas
- Next-frame prediction with Conv-LSTM
- Grad-CAM class activation visualization
- Image classification via fine-tuning with EfficientNet
- Image Classification with Vision Transformer
- Model interpretability with Integrated Gradients
- Knowledge Distillation
- Metric learning for image similarity search
- Point cloud classification with PointNet
- Few-Shot learning with Reptile
- Object Detection with RetinaNet
- Image Super-Resolution using an Efficient Sub-Pixel CNN
- Supervised Contrastive Learning
- Visualizing what convnets learn
- Pneumonia Classification on TPU

Natural language processing

- Text classification from scratch
- Sequence to sequence learning for performing number addition
- Bidirectional LSTM on IMDB
- Character-level recurrent sequence-to-sequence model
- End-to-end Masked Language Modeling with BERT
- Natural language image search with a Dual Encoder

- Diferentes casos de uso
- Atualizados frequentemente

Usuários iniciantes

- Tutoriais
- Códigos simples
- Casos de uso simples
- Conceitos básicos de TF

Computer Vision

- Image classification from scratch
- Simple MNIST convnet
- Image segmentation with a U-Net-like architecture
- 3D Image Classification from CT Scans
- OCR model for reading Captchas
- Next-frame prediction with Conv-LSTM
- Grad-CAM class activation visualization
- Image classification via fine-tuning with EfficientNet
- Image Classification with Vision Transformer
- Model interpretability with Integrated Gradients
- Knowledge Distillation
- Metric learning for image similarity search
- Point cloud classification with PointNet
- Few-Shot learning with Reptile
- Object Detection with RetinaNet
- Image Super-Resolution using an Efficient Sub-Pixel CNN
- Supervised Contrastive Learning
- Visualizing what convnets learn
- Pneumonia Classification on TPU

Natural language processing

- Text classification from scratch
- Sequence to sequence learning for performing number addition
- Bidirectional LSTM on IMDB
- Character-level recurrent sequence-to-sequence model
- End-to-end Masked Language Modeling with BERT
- Natural language image search with a Dual Encoder
- Using pre-trained word embeddings
- Semantic Similarity with BERT
- Text classification with Transformer
- Text Extraction with BERT

Usuários experientes

- Estado da arte
- Melhores práticas
- Casos de uso complexos
- Conceitos avançados de TF

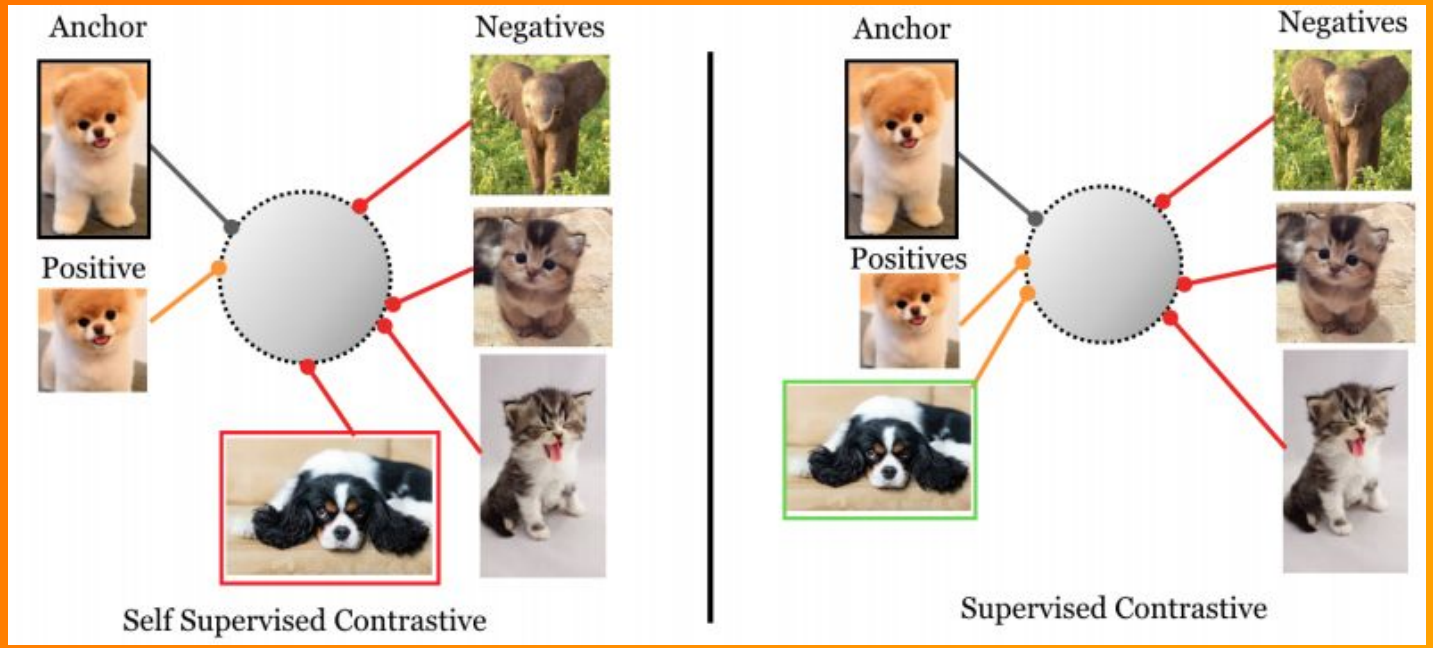
Computer Vision

- Image classification from scratch
- Simple MNIST convnet
- Image segmentation with a U-Net-like architecture
- 3D Image Classification from CT Scans
- OCR model for reading Captchas
- Next-frame prediction with Conv-LSTM
- Grad-CAM class activation visualization
- Image classification via fine-tuning with EfficientNet
- Image Classification with Vision Transformer
- Model interpretability with Integrated Gradients
- Knowledge Distillation
- Metric learning for image similarity search
- Point cloud classification with PointNet
- Few-Shot learning with Reptile
- Object Detection with RetinaNet
- Image Super-Resolution using an Efficient Sub-Pixel CNN
- Supervised Contrastive Learning
- Visualizing what convnets learn
- Pneumonia Classification on TPU

Natural language processing

- Text classification from scratch
- Sequence to sequence learning for performing number addition
- Bidirectional LSTM on IMDB
- Character-level recurrent sequence-to-sequence model
- End-to-end Masked Language Modeling with BERT
- Natural language image search with a Dual Encoder
- Using pre-trained word embeddings
- Semantic Similarity with BERT
- Text classification with Transformer
- Text Extraction with BERT

Supervised Contrastive Learning (Prannay Khosla et al.)

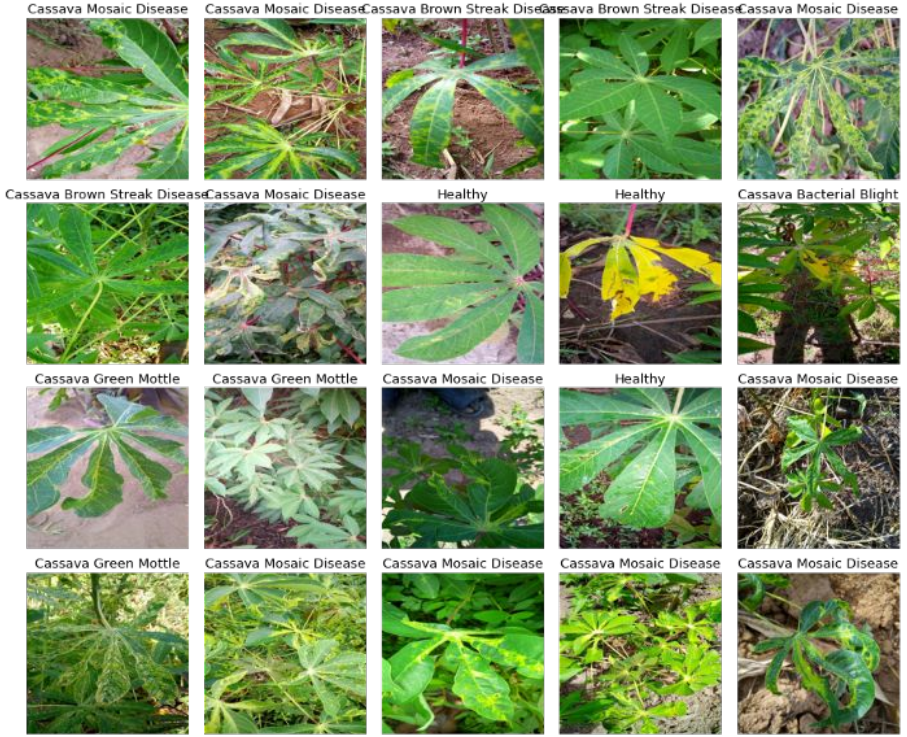


Caso de uso

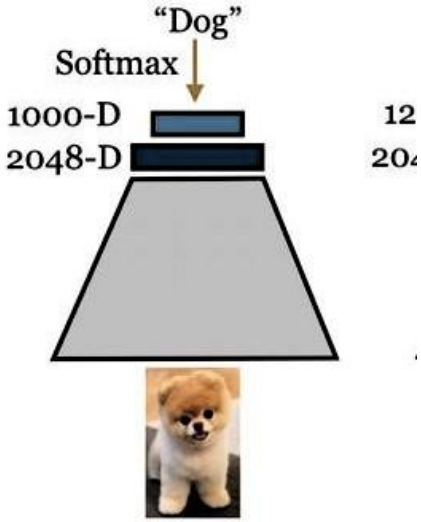
“Cassava Leaf Disease Classification” (competição no Kaggle)

Objetivo: classificar imagens de plantas (Cassava) em 5 categorias (saudáveis ou 4 doenças).

- Amostras com ruído.
- Desbalanceamento entre classes.
- Dificuldade de classificação (humana).

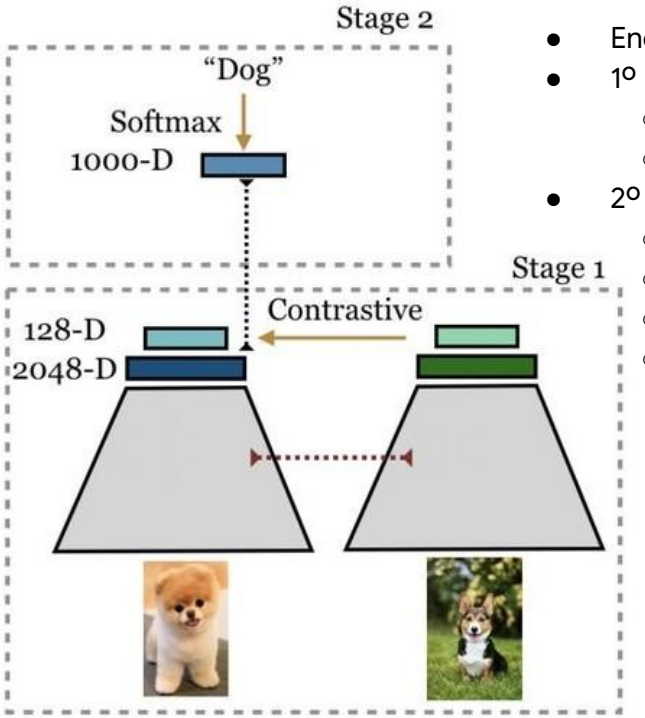


Supervised Contrastive Learning



(a) Supervised Cross Entropy

- Encoder na base (ResNet 50)
- Camadas no topo (opcionais)
- Classificador no fim (Softmax)
- Categorical cross entropy



(c) Supervised Contrastive

- Encoder na base (ResNet 50)
- 1º Estágio
  - Camada de projeção
  - Supervised contastive learning
- 2º Estágio
  - Camadas no topo (opcionais)
  - Classificador no fim (Softmax)
  - Categorical cross entropy
  - Encoder "congelado"

Encoder

```
from tensorflow.keras.applications import EfficientNetB3

def encoder_fn(input_shape):
    inputs = L.Input(shape=input_shape, name='inputs')
    base_model = EfficientNetB3(input_tensor=inputs,
                                include_top=False,
                                weights='imagenet',
                                pooling='avg')

    model = Model(inputs=inputs, outputs=base_model.outputs)
    return model
```

Camada de classificação

```
def classifier_fn(input_shape, N_CLASSES, encoder, trainable=False):
    for layer in encoder.layers:
        layer.trainable = trainable

    inputs = L.Input(shape=input_shape, name='inputs')

    features = encoder(inputs)
    features = L.Dropout(.5)(features)
    features = L.Dense(1000, activation='relu')(features)
    features = L.Dropout(.5)(features)
    outputs = L.Dense(N_CLASSES, activation='softmax', name='outputs')(features)

    model = Model(inputs=inputs, outputs=outputs)
    return model
```

Camada de projeção

```
def add_projection_head(input_shape, encoder):
    inputs = L.Input(shape=input_shape, name='inputs')
    features = encoder(inputs)
    outputs = L.Dense(128, activation='relu',
                      name='projection_head')(features)

    model = Model(inputs=inputs, outputs=outputs)
    return model
```

Supervised Contrastive Learning

```
import tensorflow_addons as tfa

class SupervisedContrastiveLoss(losses.Loss):
    def __init__(self, temperature=0.1, name=None):
        super(SupervisedContrastiveLoss, self).__init__(name=name)
        self.temperature = temperature

    def __call__(self, labels, ft_vectors, sample_weight=None):
        # Normalize feature vectors
        ft_vec_normalized = tf.math.l2_normalize(ft_vectors, axis=1)
        # Compute logits
        logits = tf.divide(tf.matmul(ft_vec_normalized, tf.transpose(ft_vec_normalized)), temperature)
        return tfa.losses.npairs_loss(tf.squeeze(labels), logits)
```

https://www.tensorflow.org/addon

Treinamento (1º estágio)

Montando o modelo

```
with strategy.scope():
    encoder = encoder_fn((None, None, 3)) # Get the encoder
    encoder_proj = add_projection_head((None, None, 3),encoder)
    # Add the projection head to the encoder

encoder_proj.compile(optimizer=optimizers.Adam(lr=3e-4),
                     loss=SupervisedContrastiveLoss(temperature=0.1))
```

Treinando

```
model.fit(x=get_dataset(TRAIN_FILENAMES, repeated=True, augment=True),
          validation_data=get_dataset(VALID_FILENAMES, ordered=True),
          steps_per_epoch=100,
          epochs=10)
```

Treinamento (2º estágio)

Montando o modelo

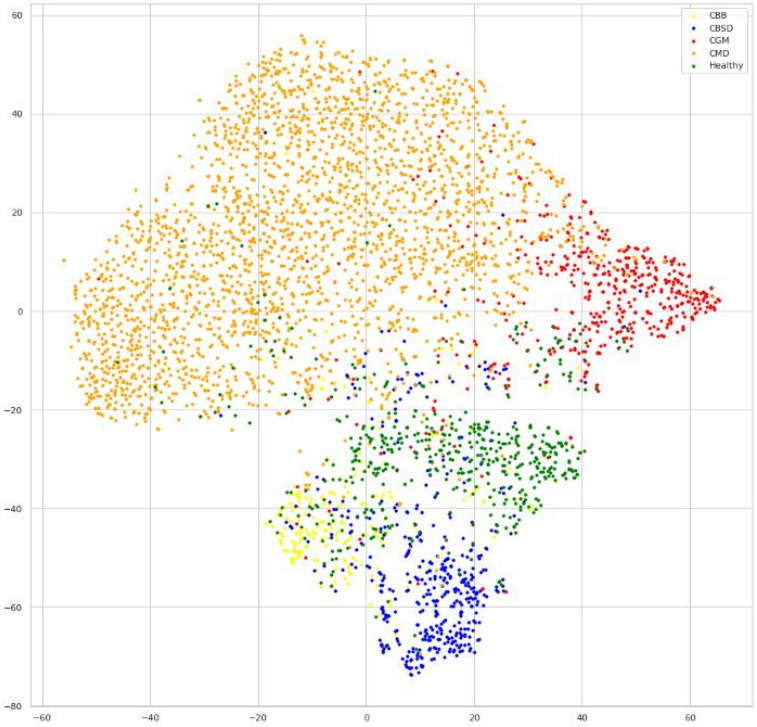
```
with strategy.scope():
    model = classifier_fn((None, None, 3), N_CLASSES,
                          encoder, # trained encoder
                          trainable=False) # with frozen weights
    model.compile(optimizer=optimizers.Adam(lr=3e-4),
                  loss=losses.SparseCategoricalCrossentropy(),
                  metrics=[metrics.SparseCategoricalAccuracy()])
```

Treinando

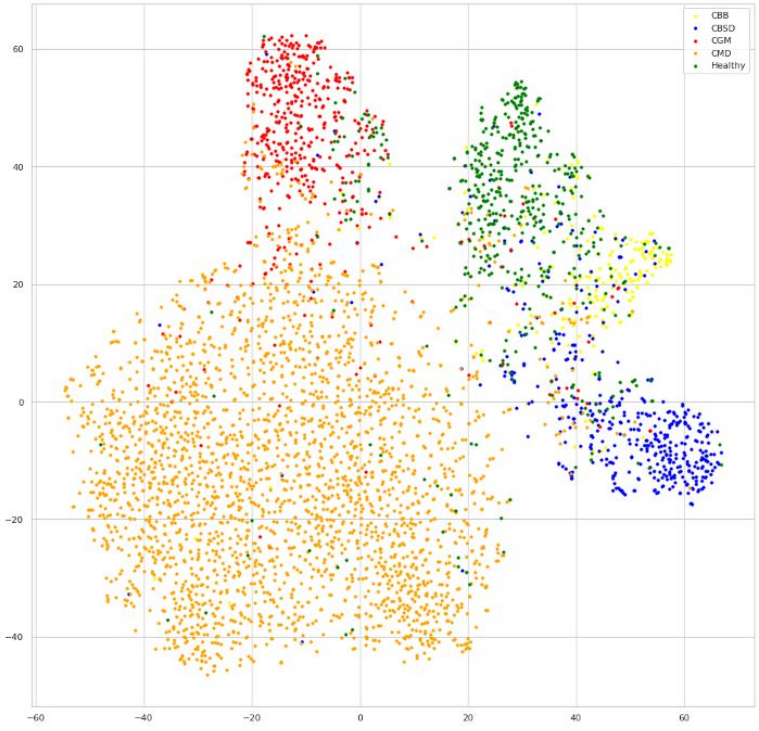
```
model.fit(x=get_dataset(TRAIN_FILENAMES, repeated=True, augment=True),
          validation_data=get_dataset(VALID_FILENAMES, ordered=True),
          steps_per_epoch=100,
          epochs=10)
```

Visualizando os resultados

Cross-entropy embedding

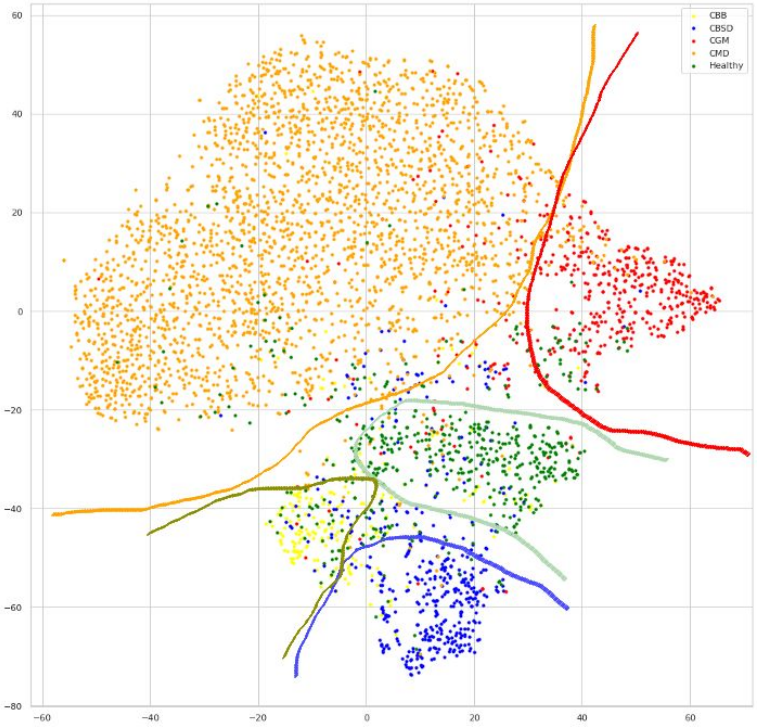


Supervised Contrastive Learning embedding

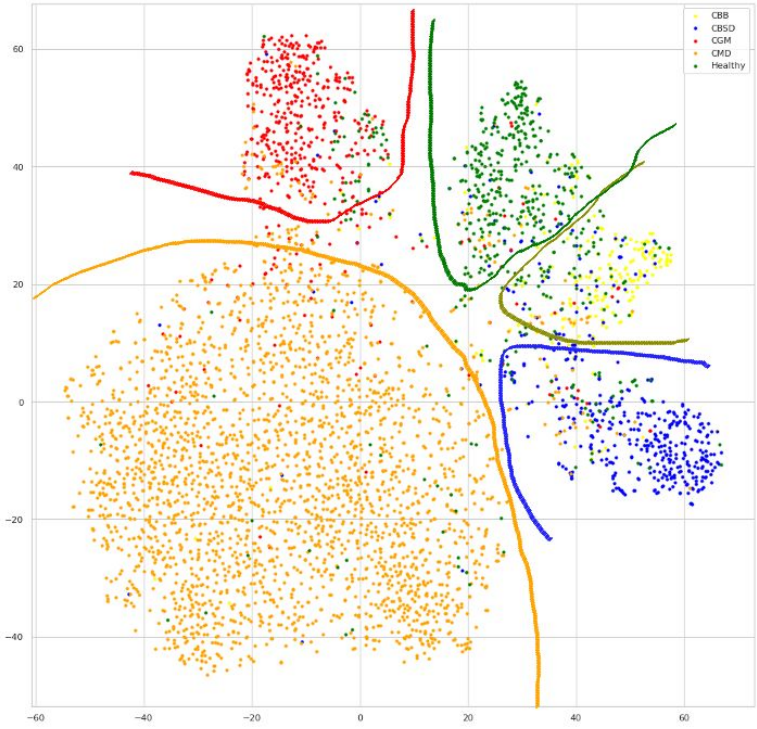


Limiares de decisão

Cross-entropy embedding



Supervised Contrastive Learning embedding



Conclusão

- Fácil implementação.
- Tempo de treino similar.
- Arquitetura flexível.
- Agrupa as amostras.
- Robustez a ruído.
- Mais generalização.

Referências

- Paper original (Prannay Khosla et al.)
- Artigo no Medium (por mim)
- Implementação @ Keras code examples (Khalid Salama)
- Implementação @ Kaggle (por mim)

TensorFlow

Obrigado

Dimitre Oliveira

Specialist Data Scientist @ Virtus, UFCG

Google Developer Expert on Machine Learning

/dimitre-oliveira-7a1a011

TensorFlow Everywhere