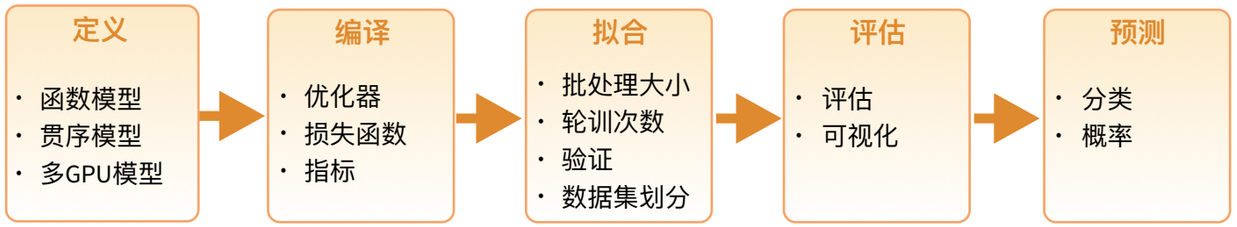
Keras 与深度学习:: 备忘单

简介

Keras 是一个为专注于快速实验而开发的高级神经网络API。它支持多个后端，包括 TensorFlow, CNTK, MXNet 和 Theano。

TensorFlow 是一个用来构建深度神经网络架构的偏底层数学计算库。keras R 包使得在R中使得 Keras 和 TensorFlow 变得非常简单。



https://keras.rstudio.com

https://www.manning.com/books/deep-learning-with-r

深度学习 “Hello, World!”

安装

keras R 包调用了 Python keras 包。 你可以直接在R中安装所有必要的依赖。

https://keras.rstudio.com/reference/install_keras.html

```
library(keras)
install_keras()
```

GPU 安装指南见 ?keras_install

本次安装需要 Anaconda 环境或者 **'r-tensorflow'** 虚拟环境。

基于MNIST 数据训练一个图像识别器

```
#输入层: 使用 MNIST 图片集
mnist <- dataset_mnist()
x_train <- mnist$train$x;  y_train <- mnist$train$y
x_test <- mnist$test$x;  y_test <- mnist$test$y

# 重塑 和 重放缩
x_train <- array_reshape(x_train, c(nrow(x_train), 784))
x_test <- array_reshape(x_test, c(nrow(x_test), 784))
x_train <- x_train / 255;  x_test <- x_test / 255

y_train <- to_categorical(y_train, 10)
y_test <- to_categorical(y_test, 10)

# 定义模型和层
model <- keras_model_sequential()
model %>%
 layer_dense(units = 256, activation = 'relu',
             input_shape = c(784)) %>%
 layer_dropout(rate = 0.4) %>%
 layer_dense(units = 128, activation = 'relu') %>%
 layer_dense(units = 10, activation = 'softmax')

# 编译(定义损失函数和优化器)
model %>% compile(
 loss = 'categorical_crossentropy',
 optimizer = optimizer_rmsprop(),
 metrics = c('accuracy')
)

# 训练 （拟合）
model %>% fit(
 x_train, y_train,
 epochs = 30, batch_size = 128,
 validation_split = 0.2
)
model %>% evaluate(x_test, y_test)
model %>% predict_classes(x_test)
```

使用 keras 模型

定义模型

keras_model() Keras 函数式模型

keras_model_sequential() 线性叠加组成的 Keras 模型

multi_gpu_model() 多GPU模型

编译模型

compile(object, optimizer, loss, metrics = NULL**)**
为模型训练添加 Keras 模型配置

拟合模型

fit(object, x = NULL, y = NULL, batch_size = NULL, epochs = 10, verbose = 1, callbacks = NULL, …**)**
以固定轮训(迭代)次数训练 Keras 模型

fit_generator() 以数据生成器的数据拟合模型

train_on_batch() test_on_batch() 单次梯度更新或通过一个批次样本评估模型

模型评估

evaluate(object, x = NULL, y = NULL, batch_size = NULL**)** 评估 Keras 模型

evaluate_generator() 在数据生成器上评估模型

预测

predict() 从Keras生成模型预测结果

predict_proba() 和 **predict_classes()**
生成输入样本的预测概率或分类概率

predict_on_batch() 返回一批样本的预测结果

predict_generator() 返回数据生成器生成的样本对应预测结果

其他模型操作

summary() 打印 Keras 模型汇总结果

export_savedmodel() 导出已保存模型

get_layer() 根据名称(唯一)或索引取出模型层

pop_layer() 移除模型末层

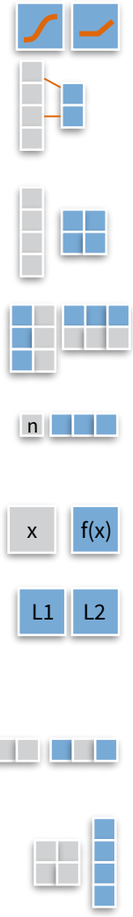
save_model_hdf5(); load_model_hdf5() 使用 HDF5 文件保存/加载模型

serialize_model(); unserialize_model()
序列化模型到一个 R 对象

clone_model() 复制一个模型实例

freeze_weights(); unfreeze_weights()
冻结或者解冻权重

核心层



layer_input() 输入层

layer_dense() 添加一个全连接层输出

layer_activation() 添加一个激活函数层到输出

layer_dropout() 添加释放层到输入

layer_reshape() 重塑输出的维度

layer_permute() 根据给定模式变更输入的维度

layer_repeat_vector() 重复 n 次输入

layer_lambda(object, f) 封装任意表达式作为一个层

layer_activity_regularization()
基于输出正则项更新损失函数

layer_masking() 用蒙版值屏蔽一个序列的特定位置的值

layer_flatten() 输入扁平化

更多层

卷积层

layer_conv_1d() 一维卷积，比如 时域卷积

layer_conv_2d_transpose() 2维转置 (反卷积)

layer_conv_2d() 二维卷积, 比如图像的空域卷积

layer_conv_3d_transpose() 3维转置 (反卷积)
layer_conv_3d() 三维卷积, 比如体积的空间卷积

layer_conv_lstm_2d() LSTM 卷积

layer_separable_conv_2d() 在深度方向上的二维可分离卷积

layer_upsampling_1d()
layer_upsampling_2d()
layer_upsampling_3d()
上采样层

layer_zero_padding_1d()
layer_zero_padding_2d()
layer_zero_padding_3d()
零填充层

layer_cropping_1d()
layer_cropping_2d()
layer_cropping_3d()
裁剪层

池化层

layer_max_pooling_1d()
layer_max_pooling_2d()
layer_max_pooling_3d()
一维到三维的最大池化

layer_average_pooling_1d()
layer_average_pooling_2d()
layer_average_pooling_3d()
一维到三维的平均池化

layer_global_max_pooling_1d()
layer_global_max_pooling_2d()
layer_global_max_pooling_3d()
全局最大池化

layer_global_average_pooling_1d()
layer_global_average_pooling_2d()
layer_global_average_pooling_3d()
全局j平均池化

激活层

layer_activation(object, activation**)**
添加一个激活函数到输出

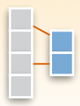
layer_activation_leaky_relu()
泄漏修正线性单元

layer_activation_parametric_relu() 参数修正线性单元

layer_activation_thresholded_relu()
阈值修正线性单元

layer_activation_elu()
指数线性单元

释放层



layer_dropout()
添加释放层到输入

layer_spatial_dropout_1d()
layer_spatial_dropout_2d()
layer_spatial_dropout_3d()
一维到三维的空间释放层

循环层

layer_simple_rnn()
将输出回馈到输入中的全连接 RNN

layer_gru()
添加门限循环单元层

layer_cudnn_gru()
基于CuDNN实现的高速GRU

layer_lstm()
添加长短期记忆模型层

layer_cudnn_lstm()
基于CuDNN实现的高速LSTM

局部连接层

layer_locally_connected_1d()
layer_locally_connected_2d()
类似卷积，但是权值不共享，比如每个批次使用不同滤波器

预处理

序列预处理

pad_sequences()
填充每个序列为同样长度 (长度最长的序列)

skipgrams()
将一个词向量下标的序列转化为一个词对

make_sampling_table()
生成词rank-based概率抽样表

文本预处理

text_tokenizer() 文本标记工具

fit_text_tokenizer() 更新分词器内部词汇

save_text_tokenizer(); load_text_tokenizer()
保存分词器到外部文件

texts_to_sequences();
texts_to_sequences_generator()
将每个文本转换成整数序列

texts_to_matrix(); sequences_to_matrix()
序列的列表转换成一个矩阵

text_one_hot() 用文本独热编码生成词索引

text_hashing_trick()
将文本转换为固定大小的散列空间的索引序列

text_to_word_sequence()
将文本转换为一系列的单词(或标记)

图像预处理

image_load() 加载一个图像到 PIL 格式。

flow_images_from_data()
flow_images_from_directory()
从图片和标签生成批增强/规范化数据/目录

image_data_generator()
生成实时的小批量图像数据

fit_image_data_generator() 根据图像数据生成器内部统计拟合样本数据

generator_next() 获取下一项

image_to_array(); image_array_resize()
image_array_save() 3维 数组表达

Keras TensorFlow

预训练模型

Keras 中的深度学习模型可以复用预训练权重。这些模型可以用于预测、特征提取和微调。

application_xception()
xception_preprocess_input()
Xception v1 模型

application_inception_v3()
inception_v3_preprocess_input()
Inception v3 模型, 在 ImageNet 上预训练权重

application_inception_resnet_v2()
inception_resnet_v2_preprocess_input()
Inception-ResNet v2 模型, 在 ImageNet 上预训练权重

application_vgg16(); application_vgg19()
VGG16 和 VGG19 模型

application_resnet50() ResNet50 模型

application_mobilenet()
mobilenet_preprocess_input()
mobilenet_decode_predictions()
mobilenet_load_model_hdf5()
MobileNet 模型架构

IMAGENET
ImageNet **是一个大型**图像标签**数据库,广泛用于深度学习**

imagenet_preprocess_input()
imagenet_decode_predictions()
预处理张量, 为ImageNet 编码一批次图片, 解码预测

回调函数

回调是被用在训练过程中的一组函数，你可以在模型训练的时候使用回调函数来获得其中的内部状态和模型统计值。

callback_early_stopping() 当监控指标没有盖上时停止训练

callback_learning_rate_scheduler() 学习速率调度器

callback_tensorboard() TensorBoard 可视化

RStudio