```
# Importacao de Bibliotecas
import keras
from keras.datasets import cifar100
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dropout, Dense
from keras.optimizers import RMSprop
```

```
# Carregar o conjunto de dados Cifar
(x_train, g_train), (x_test, g_test) = cifar100.load_data()

     Downloading data from https://www.cs.toronto.edu/~kriz/cifar-100-python.tar.gz
     169001437/169001437 [==============================] - 3s 0us/step
```

```
# Normalizacao do Conjunto de Dados
x_train = x_train / 255.0
x_test = x_test / 255.0
```

```
# Converter as classes em Categorias
y_train = to_categorical(g_train, num_classes=100)
```

```
# Criar o modelo sequencial
model = Sequential()
model.add(Conv2D(filters=32, kernel_size=(3, 3), padding='same', activation='relu', input_shape=(32, 32, 3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(filters=64, kernel_size=(3, 3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(filters=128, kernel_size=(3, 3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(filters=256, kernel_size=(3, 3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dropout(rate=0.5))
model.add(Dense(units=512, activation='relu'))
model.add(Dense(units=100, activation='softmax'))
```

```
# Compilar o modelo
model.compile(loss='categorical_crossentropy', optimizer=RMSprop(), metrics=['accuracy'])
```

```
# Treinar o modelo
history = model.fit(x_train, y_train, epochs=5, batch_size=128, validation_split=0.2)

     Epoch 1/5
     313/313 [==============================] - 169s 534ms/step - loss: 4.1717 - accuracy: 0.0577 - val_loss: 3.7085 - val_accuracy: 0.1231
     Epoch 2/5
     313/313 [==============================] - 170s 545ms/step - loss: 3.5568 - accuracy: 0.1550 - val_loss: 3.3231 - val_accuracy: 0.1971
```

```
Epoch 3/5
313/313 [==============================] - 168s 538ms/step - loss: 3.2030 - accuracy: 0.2192 - val_loss: 3.1578 - val_accuracy: 0.2231
Epoch 4/5
313/313 [==============================] - 167s 532ms/step - loss: 2.9568 - accuracy: 0.2649 - val_loss: 2.7944 - val_accuracy: 0.2905
Epoch 5/5
313/313 [==============================] - 170s 543ms/step - loss: 2.7606 - accuracy: 0.3036 - val_loss: 2.7318 - val_accuracy: 0.3144
```