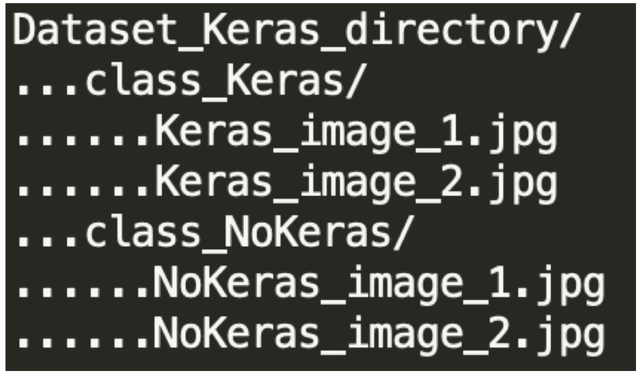
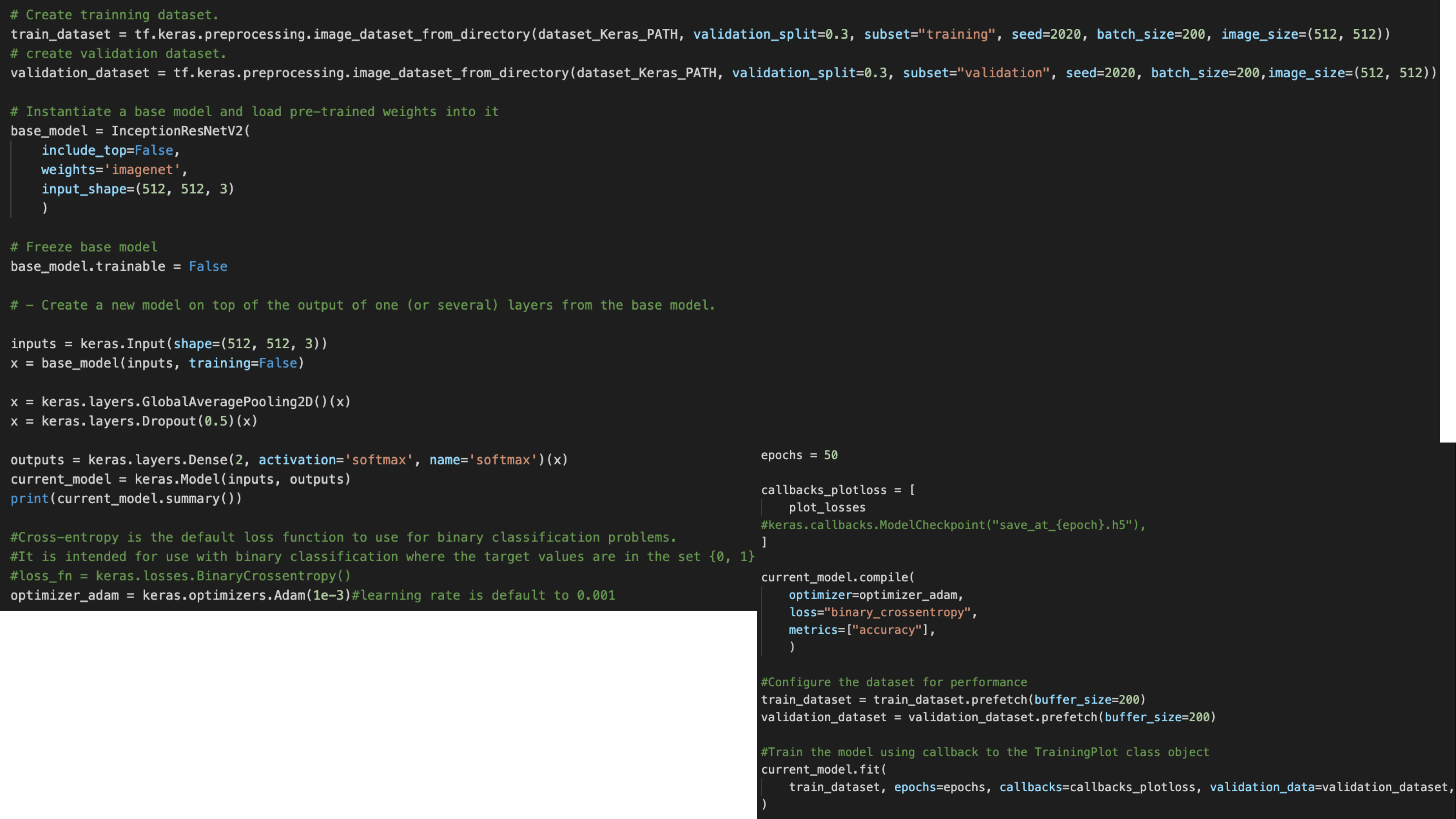
```
Found 612822 files belonging to 2 classes.
Using 428976 files for training.
Using 183846 files for validation.

Model:
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 512, 512, 3)]     0
_________________________________________________________________
inception_resnet_v2 (Functio (None, 14, 14, 1536)      54336736
_________________________________________________________________
global_average_pooling2d (Gl (None, 1536)              0
_________________________________________________________________
dropout (Dropout)            (None, 1536)              0
_________________________________________________________________
softmax (Dense)              (None, 2)                 3074
=================================================================
Total params: 54,339,810
Trainable params: 3,074
Non-trainable params: 54,336,736
_________________________________________________________________
```



```
Dataset_Keras_directory/
...class_Keras/
......Keras_image_1.jpg
......Keras_image_2.jpg
...class_NoKeras/
......NoKeras_image_1.jpg
......NoKeras_image_2.jpg
```



```python
# Create trainning dataset.
train_dataset = tf.keras.preprocessing.image_dataset_from_directory(dataset_Keras_PATH, validation_split=0.3, subset="training", seed=2020, batch_size=200, image_size=(512, 512))
# create validation dataset.
validation_dataset = tf.keras.preprocessing.image_dataset_from_directory(dataset_Keras_PATH, validation_split=0.3, subset="validation", seed=2020, batch_size=200,image_size=(512, 512))

# Instantiate a base model and load pre-trained weights into it
base_model = InceptionResNetV2(
    include_top=False,
    weights='imagenet',
    input_shape=(512, 512, 3)
    )

# Freeze base model
base_model.trainable = False

# - Create a new model on top of the output of one (or several) layers from the base model.

inputs = keras.Input(shape=(512, 512, 3))
x = base_model(inputs, training=False)

x = keras.layers.GlobalAveragePooling2D()(x)
x = keras.layers.Dropout(0.5)(x)

outputs = keras.layers.Dense(2, activation='softmax', name='softmax')(x)
current_model = keras.Model(inputs, outputs)
print(current_model.summary())

#Cross-entropy is the default loss function to use for binary classification problems.
#It is intended for use with binary classification where the target values are in the set {0, 1}
#loss_fn = keras.losses.BinaryCrossentropy()
optimizer_adam = keras.optimizers.Adam(1e-3)#learning rate is default to 0.001
```

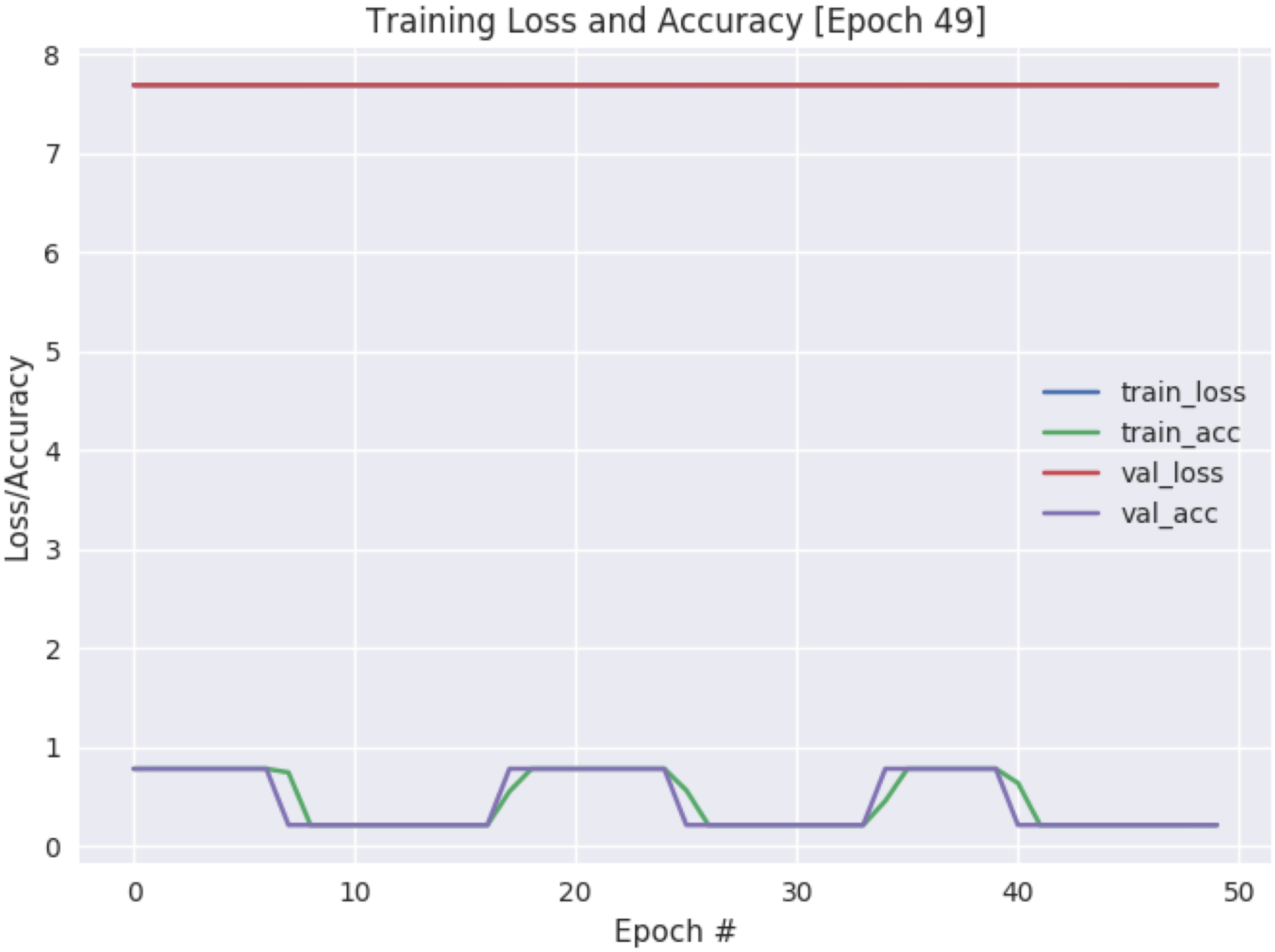
```python
epochs = 50

callbacks_plotloss = [
    plot_losses
#keras.callbacks.ModelCheckpoint("save_at_{epoch}.h5"),
]

current_model.compile(
    optimizer=optimizer_adam,
    loss="binary_crossentropy",
    metrics=["accuracy"],
    )

#Configure the dataset for performance
train_dataset = train_dataset.prefetch(buffer_size=200)
validation_dataset = validation_dataset.prefetch(buffer_size=200)

#Train the model using callback to the TrainingPlot class object
current_model.fit(
    train_dataset, epochs=epochs, callbacks=callbacks_plotloss, validation_data=validation_dataset,
)
```



Pre-train model: Inception_ResNet_V2
Added pooling, dropout, dense layer with SoftMax activation function.

Optimizer: Adam
Loss function = binary_crossentropy
Metrics = accuracy

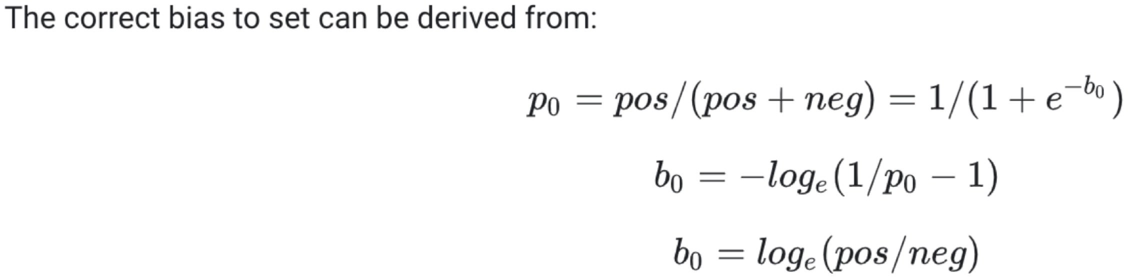
Dataset: two folder [class_kras, class_nokras]
Validation, training, => 30%, 70%
Batch_size =200,
Image size = 512, 512

Experiment epoch = 50

Problems:

1. Dataset → Imbalance class.

   [Kras: 132439/612822 => 21.6%]
   [No_Kras: 480383/612822]
   <need to adjust the initial bias, careful bias initialization help with initial convergence. >



The correct bias to set can be derived from:

$$p_0 = pos/(pos + neg) = 1/(1 + e^{-b_0})$$

$$b_0 = -log_e(1/p_0 - 1)$$

$$b_0 = log_e(pos/neg)$$

2. Small batch size. Throw OOM error on server if size > 200.
3. Evaluation metrics. Should use AUC instead of accuracy.
4. Loss function. Per tile vs. per slides? Customize loss function?
5. Use simple/small dataset to develop the scripts.
6. Server always throw OOM error, with example dataset, it works fine with google Colab.
7. GPU parallel computing

```
: I tensorflow/core/common_run Found device 0 with properties:
pciBusID: 0000:81:00.0 name: GeForce GTX 1080 Ti computeCapability: 6.1 coreClock: 1.6705GHz coreCount: 28 deviceMemorySize: 10.92GiB
deviceMemoryBandwidth: 451.17GiB/s
time/gpu/gpu_device.cc:1680]
: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero

: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1822] Adding visible gpu devices: 0

: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with Intel(R) MKL-DNN to use the following CPU instructions in
performance-critical operations:  AVX2 FMA

To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
: I tensorflow/core/platform/profile_utils/cpu_utils.cc:104] CPU Frequency: 2294685000 Hz
: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x4882f50 initialized for platform Host (this does not guarantee that XLA will be used). Devices:

: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version

: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero

: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x4885b60 initialized for platform CUDA (this does not guarantee that XLA will be used). Devices:
: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): GeForce GTX 1080 Ti, Compute Capability 6.1

: W tensorflow/core/common_runtime/bfc_allocator.cc:246] Allocator (GPU_0_bfc) ran out of memory trying to allocate 3.07GiB with freed_by_count=0. The caller
indicates that this is not a failure, but may mean that there could be performance gains if more memory were available.

: W tensorflow/core/kernels/gpu_utils.cc:49] Failed to allocate memory for convolution redzone checking; skipping this check. This is benign and only means that we
won't check cudnn for out-of-bounds reads and writes. This message will only be printed once.
```