Tools Setup

In [ ]:
```
import pathlib, os, sys, operator, re, datetime
from functools import reduce
import numpy as np
import tensorflow as tf
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
from tensorflow import keras
from keras import Model
import tensorflow_datasets as tfds
from tiny_imagenet import TinyImagenetDataset

# Enable or disable GPU
# To fully disable it, we need to hide all GPU devices from Tensorflow
# Make sure GPU is disabled for this inference part of the lab
ENABLE_GPU = True
# tf.debugging.set_log_device_placement(True)

if not ENABLE_GPU:
    tf.config.set_visible_devices([], 'GPU')

# Print Python and TF version, and where we are running
print(f'Running on Python Version: {sys.version}')
print(f'Using Tensorflow Version: {tf. __version__}')
if not tf.config.experimental.list_physical_devices("GPU"):
    print('Running on CPU')
else:
    print(f'Using GPU at: {tf.test.gpu_device_name()} (of {len(tf.config.exp
```

```
2025-09-08 12:07:10.459501: I tensorflow/core/util/port.cc:113] oneDNN custo
m operations are on. You may see slightly different numerical results due to
floating-point round-off errors from different computation orders. To turn t
hem off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-09-08 12:07:11.461606: E external/local_xla/xla/stream_executor/cuda/cu
da_fft.cc:479] Unable to register cuFFT factory: Attempting to register fact
ory for plugin cuFFT when one has already been registered
2025-09-08 12:07:11.941786: E external/local_xla/xla/stream_executor/cuda/cu
da_dnn.cc:10575] Unable to register cuDNN factory: Attempting to register fa
ctory for plugin cuDNN when one has already been registered
2025-09-08 12:07:11.945026: E external/local_xla/xla/stream_executor/cuda/cu
da_blas.cc:1442] Unable to register cuBLAS factory: Attempting to register f
actory for plugin cuBLAS when one has already been registered
2025-09-08 12:07:12.678895: I tensorflow/core/platform/cpu_feature_guard.cc:
210] This TensorFlow binary is optimized to use available CPU instructions i
n performance-critical operations.
To enable the following instructions: AVX2 AVX_VNNI FMA, in other operation
s, rebuild TensorFlow with the appropriate compiler flags.
2025-09-08 12:07:16.219884: W tensorflow/compiler/tf2tensorrt/utils/py_util
s.cc:38] TF-TRT Warning: Could not find TensorRT
```

```
Running on Python Version: 3.9.21 (main, Feb 10 2025, 00:00:00)
[GCC 11.5.0 20240719 (Red Hat 11.5.0-5)]
Using Tensorflow Version: 2.16.2
Running on CPU
```

```
2025-09-08 12:07:25.898466: I external/local_xla/xla/stream_executor/cuda/cu
da_diagnostics.cc:134] retrieving CUDA diagnostic information for host: co20
50-19.ece.iastate.edu
2025-09-08 12:07:25.898489: I external/local_xla/xla/stream_executor/cuda/cu
da_diagnostics.cc:141] hostname: co2050-19.ece.iastate.edu
2025-09-08 12:07:25.898551: I external/local_xla/xla/stream_executor/cuda/cu
da_diagnostics.cc:165] libcuda reported version is: NOT_FOUND: was unable to
find libcuda.so DSO loaded into this program
2025-09-08 12:07:25.898577: I external/local_xla/xla/stream_executor/cuda/cu
da_diagnostics.cc:169] kernel reported version is: NOT_FOUND: could not find
kernel module information in driver version file contents: "NVRM version: NV
IDIA UNIX Open Kernel Module for x86_64  580.65.06  Release Build  (root@co2
050-19.ece.iastate.edu)  Fri Aug 15 15:09:32 CDT 2025
GCC version:  gcc version 11.5.0 20240719 (Red Hat 11.5.0-5) (GCC)
"
```

Dataset Inspection

In [2]:

```python
# This cell imports our dataset.

# Original Source: https://github.com/ksachdeva/tiny-imagenet-tfds
# Class Version Source: https://github.com/duweisu/tiny-imagenet-tfds
# Setup our dataset
# ----------------------------------------------------------

tiny_imagenet_builder = TinyImagenetDataset()

# this call (download_and_prepare) will trigger the download of the dataset
# and preparation (conversion to tfrecords)
#
# This will be done only once and on next usage tfds will
# use the cached version on your host.
tiny_imagenet_builder.download_and_prepare(download_dir="~/tensorflow-datase

# class_names = tiny_imagenet_builder.info.features['label'].names
ds = tiny_imagenet_builder.as_dataset()
ds_train, ds_val = ds["train"], ds["validation"]
assert(isinstance(ds_train, tf.data.Dataset))
assert(isinstance(ds_val, tf.data.Dataset))

# Training Dataset
ds_train = ds_train.shuffle(1024).prefetch(tf.data.AUTOTUNE)

# Validation Dataset
ds_val = ds_val.shuffle(1024).prefetch(tf.data.AUTOTUNE)

# Dataset metadata
ds_info = tiny_imagenet_builder.info
```

```
WARNING:absl:You use TensorFlow DType <dtype: 'string'> in tfds.features Thi
s will soon be deprecated in favor of NumPy DTypes. In the meantime it was c
onverted to object.
```

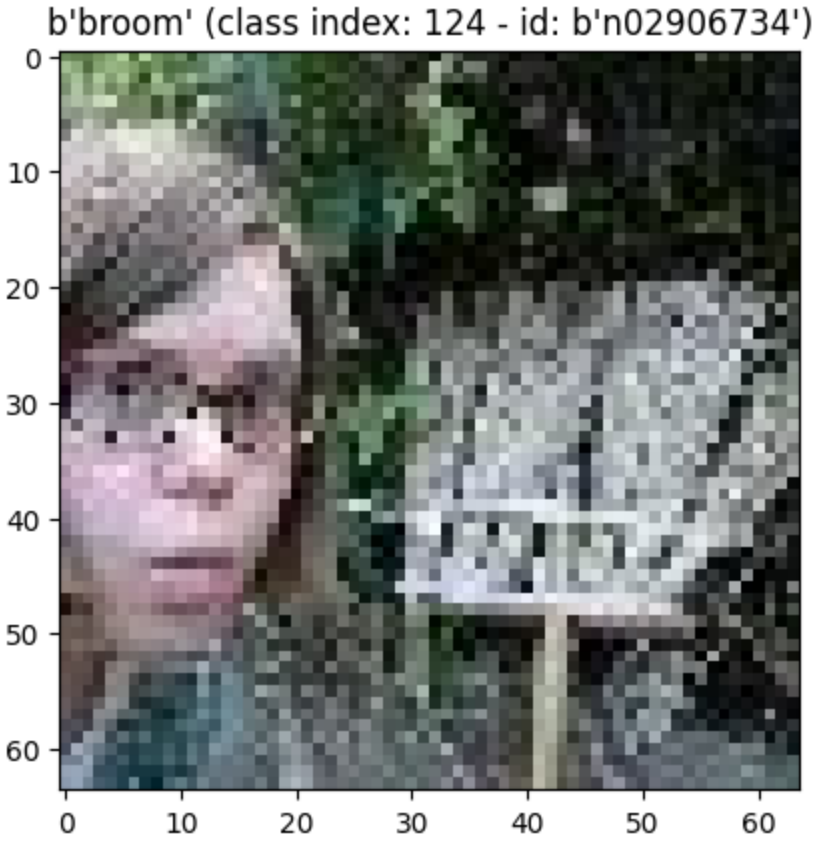
In [3]:
```python
# We need to read the "human readable" labels so we can translate with the r
# Read the labels file (words.txt)
with open(os.path.abspath('wnids.txt'), 'r') as f:
    wnids = [x.strip() for x in f]

# Map wnids to integer labels
wnid_to_label = {wnid: i for i, wnid in enumerate(wnids)}
label_to_wnid = {v: k for k, v in wnid_to_label.items()}

# Use words.txt to get names for each class
with open(os.path.abspath('words.txt'), 'r') as f:
    wnid_to_words = dict(line.split('\t') for line in f)
    for wnid, words in wnid_to_words.items():
        wnid_to_words[wnid] = [w.strip() for w in words.split(',')]

class_names = [str(wnid_to_words[wnid]) for wnid in wnids]
```

In [4]:
```python
# Helper function to get the label name
def img_class(img_data, idx=None):
    image, label, id, label_name = img_data["image"], img_data["label"], img
    # Handle batches of images correctly
    if idx != None:
        image, label, id, label_name = img_data["image"][idx], img_data["lab

    return f"{label_name} (class index: {label} - id: {id})"


# Helper function to show basic info about an image
def img_info(img, idx=None, display=True, title_apend=""):
    image = img['image']

    # Print the class
    class_str = img_class(img, idx)
    print(f"Label: {class_str}")

    # Display the image
    if display:
        plt.figure()
        plt.title(title_apend + class_str)
        # Handle batches correctly
        if image.shape.ndims > 3:
            plt.imshow(image.numpy().reshape(64, 64, 3))
        else:
            plt.imshow(image.numpy())
```

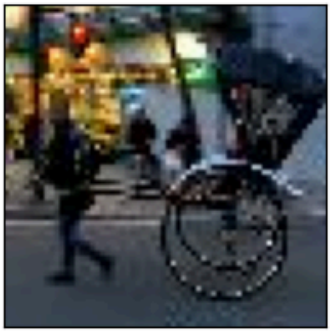
In [5]:
```python
# Print the dataset types and info
print("--- Train & Validation dataset info ---")
print(f"Train: {ds_train}")
print(f"Validation: {ds_val}")
# print(f"Dataset Info: {ds_info}") # Uncomment to print Dataset info
```

```python
print("\n--- Show an example image ---")
for example in ds_val.take(1):
    img_info(example)

print("\n Show some other examples")
tfds.show_examples(ds_val, ds_info, rows=3, cols=3)
```

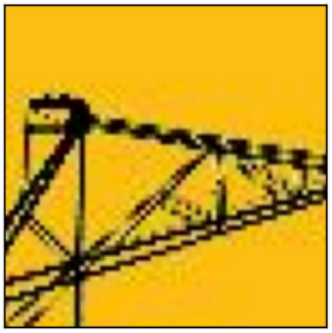
```
--- Train & Validation dataset info ---
Train: <_PrefetchDataset element_spec={'id': TensorSpec(shape=(), dtype=tf.s
tring, name=None), 'image': TensorSpec(shape=(64, 64, 3), dtype=tf.uint8, na
me=None), 'label': TensorSpec(shape=(), dtype=tf.int64, name=None), 'metadat
a': {'label_name': TensorSpec(shape=(), dtype=tf.string, name=None)}}>
Validation: <_PrefetchDataset element_spec={'id': TensorSpec(shape=(), dtype
=tf.string, name=None), 'image': TensorSpec(shape=(64, 64, 3), dtype=tf.uint
8, name=None), 'label': TensorSpec(shape=(), dtype=tf.int64, name=None), 'me
tadata': {'label_name': TensorSpec(shape=(), dtype=tf.string, name=None)}}>

--- Show an example image ---
2025-09-08 12:07:26.349885: W tensorflow/core/kernels/data/cache_dataset_op
s.cc:858] The calling iterator did not fully read the dataset being cached.
In order to avoid unexpected truncation of the dataset, the partially cached
contents of the dataset  will be discarded. This can happen if you have an i
nput pipeline similar to `dataset.cache().take(k).repeat()`. You should use
`dataset.take(k).cache().repeat()` instead.
2025-09-08 12:07:26.401154: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
2025-09-08 12:07:26.492776: W tensorflow/core/kernels/data/cache_dataset_op
s.cc:858] The calling iterator did not fully read the dataset being cached.
In order to avoid unexpected truncation of the dataset, the partially cached
contents of the dataset  will be discarded. This can happen if you have an i
nput pipeline similar to `dataset.cache().take(k).repeat()`. You should use
`dataset.take(k).cache().repeat()` instead.
2025-09-08 12:07:26.506813: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
Label: b'broom' (class index: 124 - id: b'n02906734')

 Show some other examples
```


b'broom' (class index: 124 - id: b'n02906734')
0
10
20
30
40
50
60
0
10
20
30
40
50
60

130 (130)



108 (108)

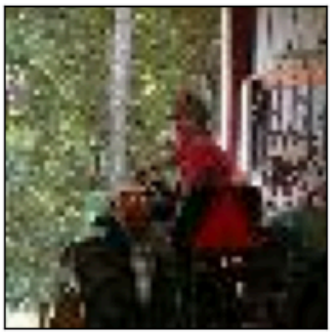


103 (103)

111 (111)

19 (19)

129 (129)

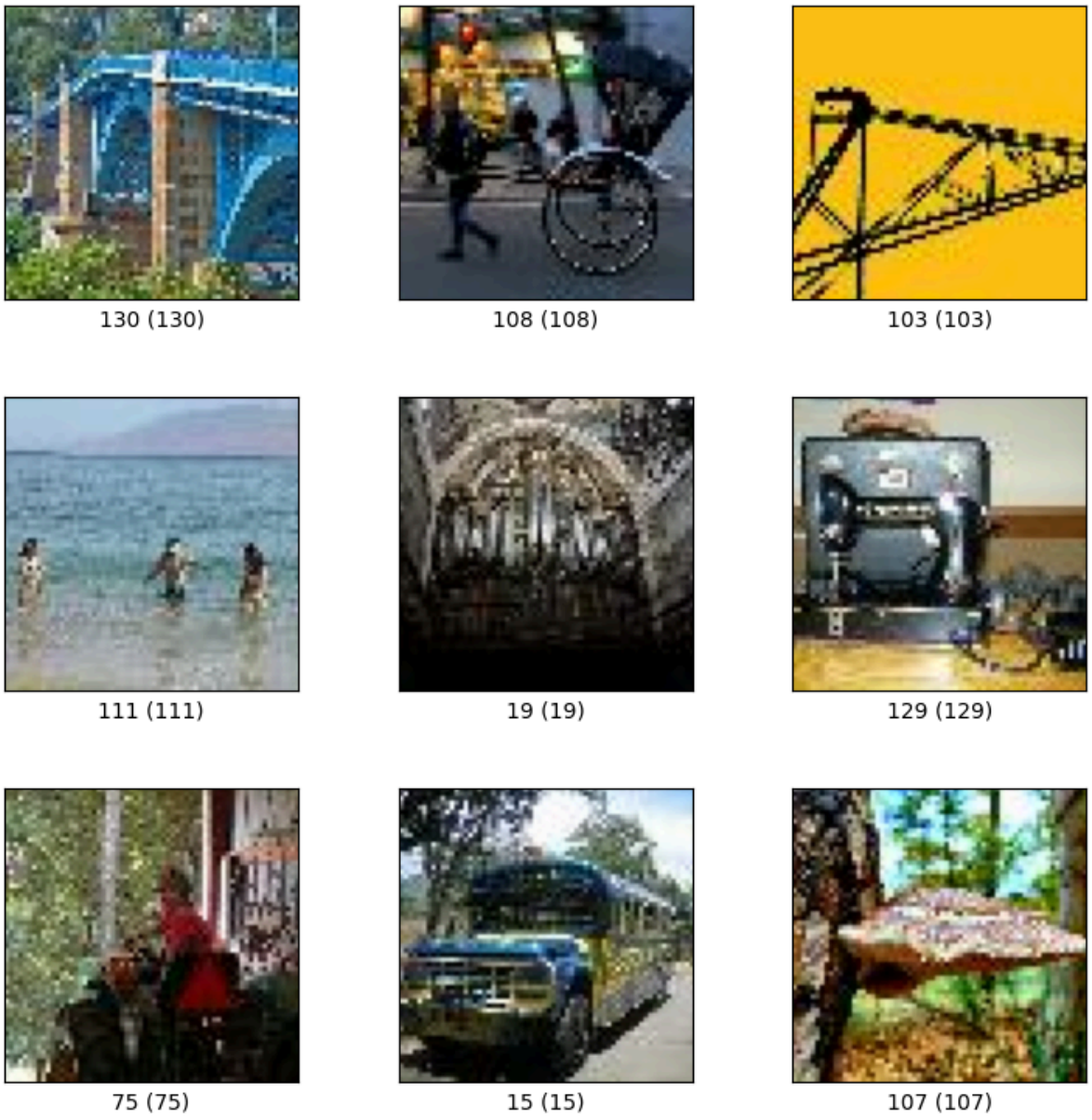


75 (75)

15 (15)

107 (107)

Out[5]:

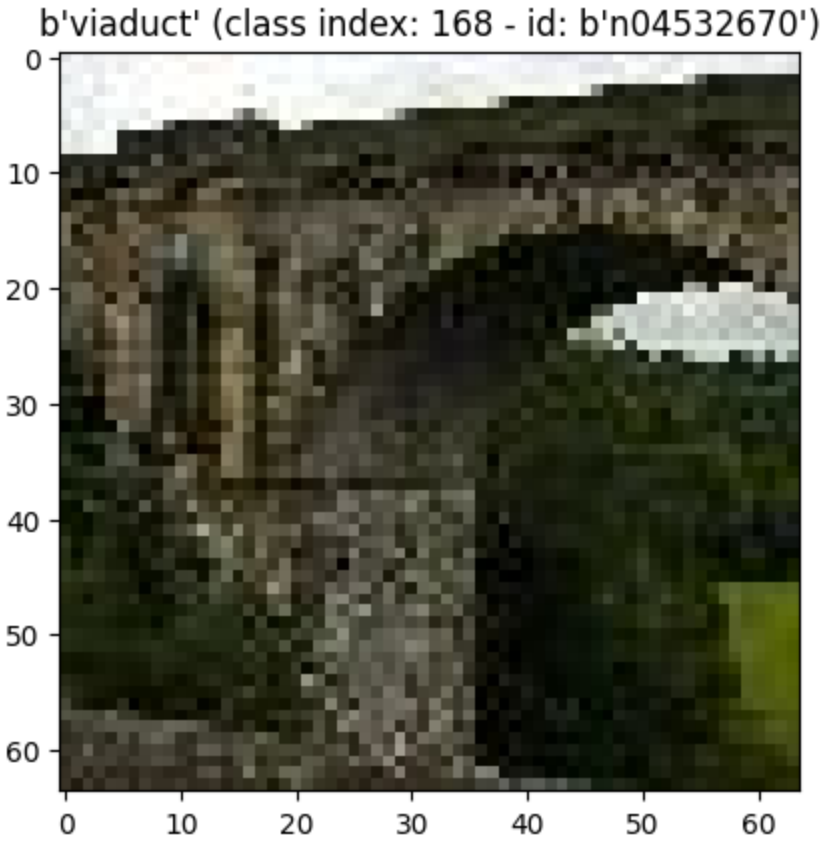


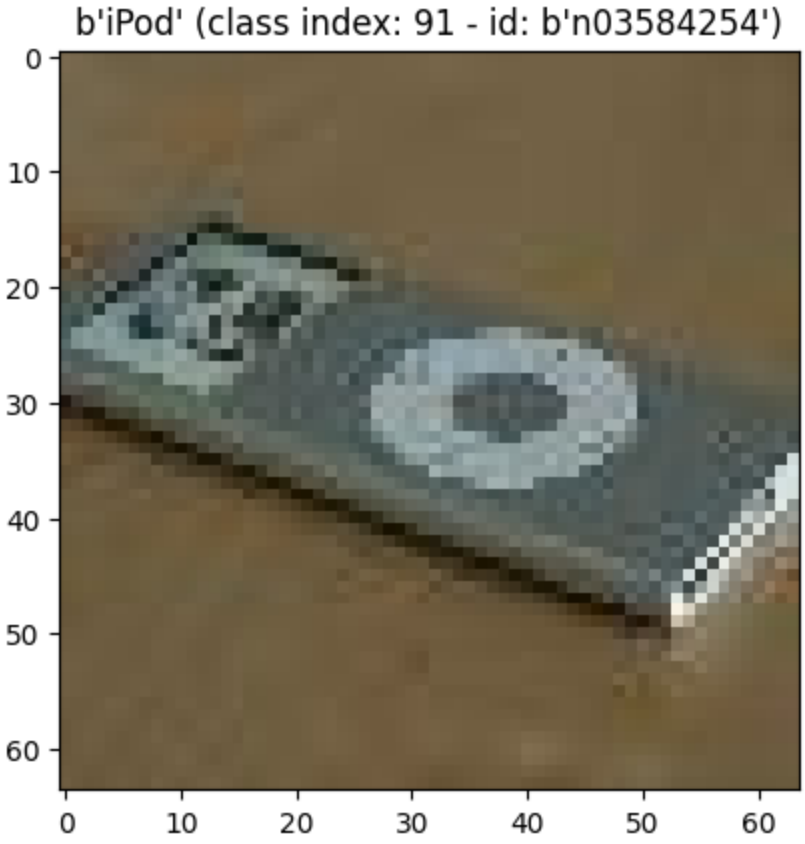
In [6]:

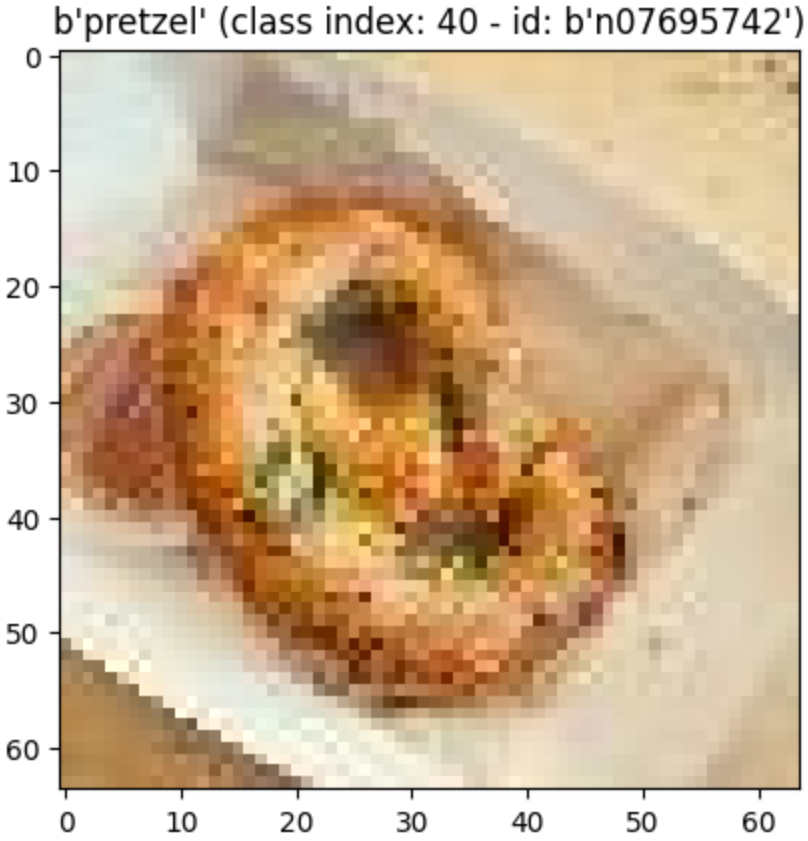
```
# TODO: Print and visualize three inputs from the validation set
#     : Print the stroage data type
#     : Print and note the dimensions of each image
#     : Print the memory required to store each image

# Sample Images
sample_imgs = []
for index, img_data in enumerate(ds_val.take(3)):
    sample_imgs.append(img_data)
    image, label, id, label_name = img_data["image"], img_data["label"], img

    print(f'\n--- Image {index} ---')
    # TODO: Your Code Here
    img_info(img_data)
    print(f"Data Type - {image.dtype}")
    print(f"Dimensions - {image.shape}")
    print("uint8 = 1 byte\n64 x 64 x 3 * 1 byte = 12228 bytes")
    # See example usage: https://github.com/duweisu/tiny-imagenet-tfds
```

```
--- Image 0 ---
Label: b'viaduct' (class index: 168 - id: b'n04532670')
Data Type - <dtype: 'uint8'>
Dimensions - (64, 64, 3)
uint8 = 1 byte
64 x 64 x 3 * 1 byte = 12228 bytes

--- Image 1 ---
Label: b'iPod' (class index: 91 - id: b'n03584254')
Data Type - <dtype: 'uint8'>
Dimensions - (64, 64, 3)
uint8 = 1 byte
64 x 64 x 3 * 1 byte = 12228 bytes

--- Image 2 ---
Label: b'pretzel' (class index: 40 - id: b'n07695742')
Data Type - <dtype: 'uint8'>
Dimensions - (64, 64, 3)
uint8 = 1 byte
64 x 64 x 3 * 1 byte = 12228 bytes
```

```
2025-09-08 12:07:27.232384: W tensorflow/core/kernels/data/cache_dataset_op
s.cc:858] The calling iterator did not fully read the dataset being cached.
In order to avoid unexpected truncation of the dataset, the partially cached
contents of the dataset  will be discarded. This can happen if you have an i
nput pipeline similar to `dataset.cache().take(k).repeat()`. You should use
`dataset.take(k).cache().repeat()` instead.
2025-09-08 12:07:27.238809: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```







In [7]:
```
# TODO: Export each of the three inputs to a binary file which will be used
# NOTE: First flatten the array (ex: 4D --> 1D). So 64*64*3 = 12288 element
```

```
# Make a directory for our image data
img_dir = os.path.abspath('img_data')
pathlib.Path(img_dir).mkdir(exist_ok=True)

# Create a metadata file
metadata_file = open(os.path.join(img_dir, f'metadata.txt'), 'w')
metadata_file.write(f'Number\t\tDims\t\tClass Data\n')

# Export each image
for index, img_data in enumerate(sample_imgs):
    img_file = open(os.path.join(img_dir, f'image_{index}.bin'), 'wb')

    # TODO: Your Code Here
    arr = img_data["image"].numpy()
    label = img_data["label"].numpy()
    flat_arr = arr.flatten()
    img_file.write(flat_arr.tobytes())
    img_file.close()
    metadata_file.write(f"{index}\t\t{image.shape}\t\t{label}\n")
```

Model Loading and Inference

In [8]:
```
# TODO: Load the model
# Now we will load the H5 model! Please make sure the h5 model file is prese
# You can download this from the Canvas Page and place it in the same direct

# model_path = os.path.abspath(""/home/<NETID>/path/to/your/lab1/CNN_TinyIma
model_path = os.path.abspath("CNN_TinyImageNet.h5")

# TODO: Your Code Here
model = tf.keras.models.load_model(model_path)
model.summary()
```

```
/home/buenting/cpre_5870/lab1/Lab1/lab1_venv/lib64/python3.9/site-packages/k
eras/src/layers/convolutional/base_conv.py:113: UserWarning: Do not pass an
`input_shape`/`input_dim` argument to a layer. When using Sequential models,
prefer using an `Input(shape)` object as the first layer in the model instea
d.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
/home/buenting/cpre_5870/lab1/Lab1/lab1_venv/lib64/python3.9/site-packages/k
eras/src/optimizers/base_optimizer.py:86: UserWarning: Argument `decay` is n
o longer supported and will be ignored.
  warnings.warn(
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to
be built. `model.compile_metrics` will be empty until you train or evaluate
the model.
WARNING:absl:Error in loading the saved optimizer state. As a result, your m
odel is starting with a freshly initialized optimizer.
```

Model: "sequential"

Layer (type)	Output Shape	Par
conv2d (Conv2D)	(None, 60, 60, 32)	2
conv2d_1 (Conv2D)	(None, 56, 56, 32)	25
max_pooling2d (MaxPooling2D)	(None, 28, 28, 32)	
conv2d_2 (Conv2D)	(None, 26, 26, 64)	18
conv2d_3 (Conv2D)	(None, 24, 24, 64)	36
max_pooling2d_1 (MaxPooling2D)	(None, 12, 12, 64)	
conv2d_4 (Conv2D)	(None, 10, 10, 64)	36
conv2d_5 (Conv2D)	(None, 8, 8, 128)	73
max_pooling2d_2 (MaxPooling2D)	(None, 4, 4, 128)	
flatten (Flatten)	(None, 2048)	
dense (Dense)	(None, 256)	524
dense_1 (Dense)	(None, 200)	51

Total params: 770,218 (2.94 MB)

Trainable params: 770,216 (2.94 MB)

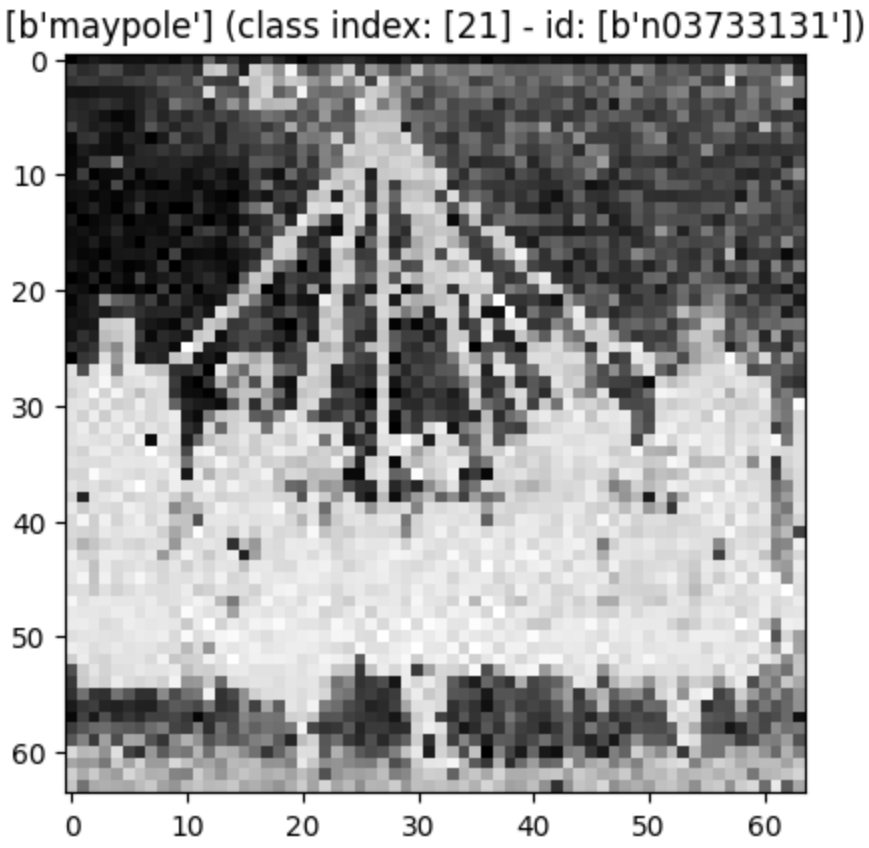
Non-trainable params: 0 (0.00 B)

Optimizer params: 2 (12.00 B)

In [9]:
```
# Running infrence on our model
# We can run an infrence of our model by doing the following (we are doing b
for example in ds_train.batch(1).take(1):
    img_info(example)
    # Make a prediction
    pred = model.predict(tf.cast(example["image"], tf.float32)/255.0)
    # print(f'Raw 200 Class Weighted Prediction:\n{pred}') # Uncomment to se

    # What is out best guess?
    best_guess = tf.math.argmax(pred, axis=1).numpy() # Our output is 200 we
    print(f'Best Guess [class index]: {class_names[best_guess[0]]} [{best_gu
    print(f'Best Guess Confidence (percent / 1.0): {pred[0][best_guess]}')

    # What are our top 15 guesses?
    top_15 = tf.math.top_k(pred, k=15)
    print(f'Top 15 Guesses (class index): {[f"{class_names[idx]} [{idx}]" fo
    print(f'Top 15 Guesses Confidence (percent / 1.0): {top_15.values}')
```

```
2025-09-08 12:07:27.843166: W tensorflow/core/kernels/data/cache_dataset_op
s.cc:858] The calling iterator did not fully read the dataset being cached.
In order to avoid unexpected truncation of the dataset, the partially cached
contents of the dataset  will be discarded. This can happen if you have an i
nput pipeline similar to `dataset.cache().take(k).repeat()`. You should use
`dataset.take(k).cache().repeat()` instead.
```

```
Label: [b'maypole'] (class index: [21] - id: [b'n03733131'])
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 264ms/step
Best Guess [class index]: ['tabby', 'tabby cat'] [66]
Best Guess Confidence (percent / 1.0): [0.12290254]
Top 15 Guesses (class index): ["['tabby', 'tabby cat'] [66]", "['nail'] [2
9]", "['walking stick', 'walkingstick', 'stick insect'] [181]", "['fountai
n'] [140]", "['suspension bridge'] [119]", "['stopwatch', 'stop watch'] [13
7]", "['American alligator', 'Alligator mississipiensis'] [160]", "['steel a
rch bridge'] [130]", "['chain'] [171]", "['rocking chair', 'rocker'] [3]", "
['koala', 'koala bear', 'kangaroo bear', 'native bear', 'Phascolarctos ciner
eus'] [68]", "['magnetic compass'] [191]", "['maypole'] [21]", "['birdhous
e'] [72]", "['Egyptian cat'] [0]"]
Top 15 Guesses Confidence (percent / 1.0): [[0.12290254 0.09233914 0.0604626
7 0.05863593 0.05133322 0.04556591
  0.0418246  0.03907003 0.03407361 0.02940531 0.02923596 0.0203824
  0.0202339  0.01993833 0.01878766]]
2025-09-08 12:07:28.178480: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```



In [10]:
```
# TODO: Run infrence for our previous 3 sample images

# TODO: Your Code Here

# for index, img_data in enumerate(sample_imgs):

#     pred = model.predict(tf.cast(img_data["image"], tf.float32)/255.0)
#     sample_predictions.append(pred)
#     best_guess = tf.math.argmax(sample_predictions[index], axis=1).numpy()
#     label = sample_predictions[index]["label"].numpy()
#     print(f"{best_guess}\t\t{pred[0][best_guess]}\t\t")
```

```
# Stack all 3 sample images into a stack for the prediction model (3, 64, 64
imgs = tf.stack([tf.cast(d["image"], tf.float32) / 255.0 for d in sample_img

preds = model.predict(imgs)

print(f"Best guess\t\tAssoc. Conf.\t\tActual Class\n")
for i, pred in enumerate(preds):
    best_guess = np.argmax(pred)
    confidence = pred[best_guess]
    label = sample_imgs[i]["label"].numpy()

    print(f"{best_guess}\t\t{confidence}\t\t{label}")
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 94ms/step
Best guess              Assoc. Conf.            Actual Class

168             0.4106539785861969              168
26              0.2709677219390869              91
40              0.4427526593208313              40
```

In [ ]:

```
# # TODO: Calculate the Top-1, Top-5, and Top-10 Accuracy of the validation
total = acc_top1 = acc_top5 = acc_top10 = 0
total = ds_val.cardinality().numpy()
# # TODO: Your Code Here
for images in ds_val.batch(32):
    # Predict the classes for each image, in a batch of 32 images at a time
    preds = model.predict(tf.cast(images["image"], tf.float32)/255.0, verbos
    labels = images["label"].numpy()

    for i, pred in enumerate(preds):
        top_1 = np.argmax(pred)
        if top_1 == labels[i]:
            acc_top1 += 1

        # top_5 = tf.math.top_k(pred, n=5)
        # for j in range(5):
        #     if top_5[j] == labels[i]:
        #         acc_top5 += 1
        #         break

        # top_10 = tf.math.top_k(pred, n=10)
        # for j in range(10):
        #     if top_10[j] == labels[i]:
        #         acc_top10 += 1
        #         break

        top_5 = tf.math.top_k(pred, k=5).indices.numpy()
        if labels[i] in top_5:
            acc_top5 += 1

        top_10 = tf.math.top_k(pred, k=10).indices.numpy()
        if labels[i] in top_10:
            acc_top10 += 1

print(f"Top 1 Accuracy: {acc_top1/total}\n")
```

```
print(f"Top 5 Accuracy: {acc_top5/total}\n")
print(f"Top 10 Accuracy: {acc_top10/total}\n")
```

```
Top 1 Accuracy: 0.243

Top 5 Accuracy: 0.494

Top 10 Accuracy: 0.6174
```

```
2025-09-08 12:07:45.973865: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```

In [12]:

```
# # TODO: Print all of the possible classes of the dataset

# # TODO: Your Code Here
train_labels = set()
for batch in ds_train.batch(32):
    train_labels.update(batch["label"].numpy().tolist())

val_labels = set()
for batch in ds_train.batch(32):
    val_labels.update(batch["label"].numpy().tolist())

print("Training classes:", len(train_labels))
print("Validation classes:", len(val_labels))

for i in range(200):
    print({class_names[i],})
```

```
2025-09-08 12:07:48.673151: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```

```
Training classes: 200
Validation classes: 200
{"['Egyptian cat']"}
{"['reel']"}
{"['volleyball']"}
{"['rocking chair', 'rocker']"}
{"['lemon']"}
{"['bullfrog', 'Rana catesbeiana']"}
{"['basketball']"}
{"['cliff', 'drop', 'drop-off']"}
{"['espresso']"}
{'[\'plunger\', "plumber\'s helper"]'}
{"['parking meter']"}
{"['German shepherd', 'German shepherd dog', 'German police dog', 'alsatia
n']"}
{"['dining table', 'board']"}
{"['monarch', 'monarch butterfly', 'milkweed butterfly', 'Danaus plexippu
s']"}
{"['brown bear', 'bruin', 'Ursus arctos']"}
{"['school bus']"}
{"['pizza', 'pizza pie']"}
{"['guinea pig', 'Cavia cobaya']"}
{"['umbrella']"}
{"['organ', 'pipe organ']"}
{"['oboe', 'hautboy', 'hautbois']"}
{"['maypole']"}
{"['goldfish', 'Carassius auratus']"}
{"['potpie']"}
{"['hourglass']"}
{"['seashore', 'coast', 'seacoast', 'sea-coast']"}
{"['computer keyboard', 'keypad']"}
{"['Arabian camel', 'dromedary', 'Camelus dromedarius']"}
{"['ice cream', 'icecream']"}
{"['nail']"}
{"['space heater']"}
{"['cardigan']"}
{"['baboon']"}
{"['snail']"}
{"['coral reef']"}
{"['albatross', 'mollymawk']"}
{'[\'spider web\', "spider\'s web"]'}
{"['sea cucumber', 'holothurian']"}
{"['backpack', 'back pack', 'knapsack', 'packsack', 'rucksack', 'haversac
k']"}
{"['Labrador retriever']"}
{"['pretzel']"}
{"['king penguin', 'Aptenodytes patagonica']"}
{"['sulphur butterfly', 'sulfur butterfly']"}
{"['tarantula']"}
{"['lesser panda', 'red panda', 'panda', 'bear cat', 'cat bear', 'Ailurus fu
lgens']"}
{"['pop bottle', 'soda bottle']"}
{"['banana']"}
{"['sock']"}
{"['cockroach', 'roach']"}
{"['projectile', 'missile']"}
```

{"['beer bottle']"}
{"['mantis', 'mantid']"}
{"['freight car']"}
{"['guacamole']"}
{"['remote control', 'remote']"}
{"['European fire salamander', 'Salamandra salamandra']"}
{"['lakeside', 'lakeshore']"}
{"['chimpanzee', 'chimp', 'Pan troglodytes']"}
{"['pay-phone', 'pay-station']"}
{"['fur coat']"}
{"['alp']"}
{"['lampshade', 'lamp shade']"}
{"['torch']"}
{"['abacus']"}
{"['moving van']"}
{"['barrel', 'cask']"}
{"['tabby', 'tabby cat']"}
{"['goose']"}
{"['koala', 'koala bear', 'kangaroo bear', 'native bear', 'Phascolarctos cin
ereus']"}
{"['bullet train', 'bullet']"}
{"['CD player']"}
{"['teapot']"}
{"['birdhouse']"}
{"['gazelle']"}
{'[\'academic gown\', \'academic robe\', "judge\'s robe"]'}
{"['tractor']"}
{"['ladybug', 'ladybeetle', 'lady beetle', 'ladybird', 'ladybird beetle']"}
{"['miniskirt', 'mini']"}
{"['golden retriever']"}
{"['triumphal arch']"}
{"['cannon']"}
{"['neck brace']"}
{"['sombrero']"}
{"['gasmask', 'respirator', 'gas helmet']"}
{"['candle', 'taper', 'wax light']"}
{"['desk']"}
{"['frying pan', 'frypan', 'skillet']"}
{"['bee']"}
{"['dam', 'dike', 'dyke']"}
{"['spiny lobster', 'langouste', 'rock lobster', 'crawfish', 'crayfish', 'se
a crawfish']"}
{"['police van', 'police wagon', 'paddy wagon', 'patrol wagon', 'wagon', 'bl
ack Maria']"}
{"['iPod']"}
{"['punching bag', 'punch bag', 'punching ball', 'punchball']"}
{"['beacon', 'lighthouse', 'beacon light', 'pharos']"}
{"['jellyfish']"}
{"['wok']"}
{'["potter\'s wheel"]'}
{"['sandal']"}
{"['pill bottle']"}
{"['butcher shop', 'meat market']"}
{"['slug']"}
{"['hog', 'pig', 'grunter', 'squealer', 'Sus scrofa']"}
{"['cougar', 'puma', 'catamount', 'mountain lion', 'painter', 'panther', 'Fe

```
lis concolor']"}
{"['crane']"}
{"['vestment']"}
{'[\'dragonfly\', \'darning needle\', "devil\'s darning needle", \'sewing ne
edle\', \'snake feeder\', \'snake doctor\', \'mosquito hawk\', \'skeeter haw
k\']'}
{"['cash machine', 'cash dispenser', 'automated teller machine', 'automatic
teller machine', 'automated teller', 'automatic teller', 'ATM']"}
{"['mushroom']"}
{"['jinrikisha', 'ricksha', 'rickshaw']"}
{"['water tower']"}
{"['chest']"}
{"['snorkel']"}
{"['sunglasses', 'dark glasses', 'shades']"}
{"['fly']"}
{"['limousine', 'limo']"}
{"['black stork', 'Ciconia nigra']"}
{"['dugong', 'Dugong dugon']"}
{"['sports car', 'sport car']"}
{"['water jug']"}
{"['suspension bridge']"}
{"['ox']"}
{"['ice lolly', 'lolly', 'lollipop', 'popsicle']"}
{"['turnstile']"}
{"['Christmas stocking']"}
{"['broom']"}
{"['scorpion']"}
{"['wooden spoon']"}
{"['picket fence', 'paling']"}
{"['rugby ball']"}
{"['sewing machine']"}
{"['steel arch bridge']"}
{"['Persian cat']"}
{"['refrigerator', 'icebox']"}
{"['barn']"}
{"['apron']"}
{"['Yorkshire terrier']"}
{"['swimming trunks', 'bathing trunks']"}
{"['stopwatch', 'stop watch']"}
{"['lawn mower', 'mower']"}
{"['thatch', 'thatched roof']"}
{"['fountain']"}
{"['black widow', 'Latrodectus mactans']"}
{"['bikini', 'two-piece']"}
{"['plate']"}
{"['teddy', 'teddy bear']"}
{"['barbershop']"}
{"['confectionery', 'confectionary', 'candy store']"}
{"['beach wagon', 'station wagon', 'wagon', 'estate car', 'beach waggon', 's
tation waggon', 'waggon']"}
{"['scoreboard']"}
{"['orange']"}
{"['flagpole', 'flagstaff']"}
{"['American lobster', 'Northern lobster', 'Maine lobster', 'Homarus america
nus']"}
{"['trolleybus', 'trolley coach', 'trackless trolley']"}
```

{"['drumstick']"}
{"['dumbbell']"}
{"['brass', 'memorial tablet', 'plaque']"}
{"['bow tie', 'bow-tie', 'bowtie']"}
{"['convertible']"}
{"['bighorn', 'bighorn sheep', 'cimarron', 'Rocky Mountain bighorn', 'Rocky Mountain sheep', 'Ovis canadensis']"}
{"['orangutan', 'orang', 'orangutang', 'Pongo pygmaeus']"}
{"['American alligator', 'Alligator mississipiensis']"}
{"['centipede']"}
{"['syringe']"}
{"['go-kart']"}
{"['brain coral']"}
{"['sea slug', 'nudibranch']"}
{"['cliff dwelling']"}
{"['mashed potato']"}
{"['viaduct']"}
{"['military uniform']"}
{"['pomegranate']"}
{"['chain']"}
{"['kimono']"}
{"['comic book']"}
{"['trilobite']"}
{"['bison']"}
{"['pole']"}
{"['boa constrictor', 'Constrictor constrictor']"}
{"['poncho']"}
{"['bathtub', 'bathing tub', 'bath', 'tub']"}
{"['grasshopper', 'hopper']"}
{"['walking stick', 'walkingstick', 'stick insect']"}
{"['Chihuahua']"}
{"['tailed frog', 'bell toad', 'ribbed toad', 'tailed toad', 'Ascaphus trui']"}
{"['lion', 'king of beasts', 'Panthera leo']"}
{"['altar']"}
{"['obelisk']"}
{"['beaker']"}
{"['bell pepper']"}
{"['bannister', 'banister', 'balustrade', 'balusters', 'handrail']"}
{"['bucket', 'pail']"}
{"['magnetic compass']"}
{"['meat loaf', 'meatloaf']"}
{"['gondola']"}
{"['standard poodle']"}
{"['acorn']"}
{"['lifeboat']"}
{"['binoculars', 'field glasses', 'opera glasses']"}
{"['cauliflower']"}
{"['African elephant', 'Loxodonta africana']"}

2025-09-08 12:07:48.944399: W tensorflow/core/framework/local_rendezvous.cc:404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence

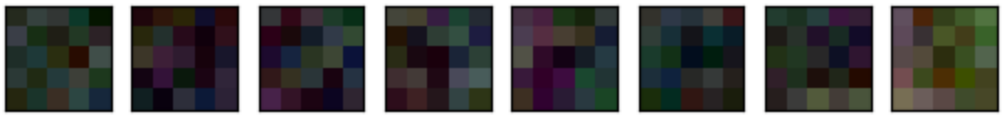
Model Exploration

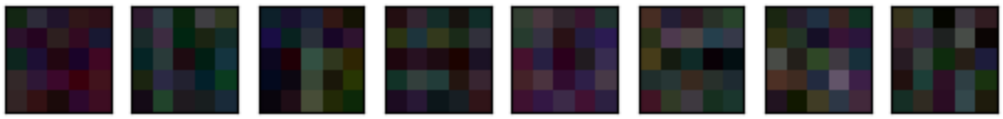
In [13]:
```
# TODO: Visualize the model in Netron (https://netron.app/) and include an i
# tf.keras.utils.plot_model(model, "model.png", show_shapes=True, show_dtype
```

In [14]:
```
# We can view the layer weights. Here we consider them as images of feature
# TODO: Visualize the 2 convolutional layers filter sets (weights) (one at t

# TODO: Your Code Here
conv_layers = [l for l in model.layers if isinstance(l, tf.keras.layers.Conv
first_conv = conv_layers[0]
last_conv = conv_layers[-2]

def describe_layer(layer):
    W, b = layer.get_weights()
    n_params = W.size + b.size
    mem_bytes = W.nbytes + b.nbytes
    print(f"\nLayer: {layer.name} ({layer.__class__.__name__})")
    print(f"Weights shape: {W.shape}, Bias shape: {b.shape}")
    print(f"dtype: {W.dtype}")
    print(f"Params: {n_params:,}")
    print(f"Memory: {mem_bytes} bytes")
    return W,b

W1, b1 = describe_layer(first_conv)
W2, b2 = describe_layer(last_conv)

def normalize(x):
    x = x - x.min()
    m = x.min()
    return x / (m if m != 0 else 1.0)

num_show1 = min(16, W1.shape[-1])
cols = 8
rows = int(num_show1 / cols)
plt.figure()
for i in range(num_show1):
    f = W1[:, :, :, i]
    f_img = normalize(f)
    ax = plt.subplot(rows, cols, i+1)
    ax.imshow(f_img)
    ax.set_xticks([]); ax.set_yticks([])
plt.suptitle(f"{first_conv.name} first {num_show1} filters")
plt.show()

num_show2 = min(16, W2.shape[-1])
rows = int(num_show2 / cols)
plt.figure()
for i in range(num_show2):
    k = W2[:, :, :, i]
    energy = np.sqrt((k**2).sum(axis=2))
    e_img = normalize(energy)
    ax = plt.subplot(rows, cols, i+1)
    ax.imshow(e_img)
    ax.set_xticks([]); ax.set_yticks([])
plt.suptitle(f"{last_conv.name} first {num_show2} filters")
plt.show()
```

```
Layer: conv2d (Conv2D)
Weights shape: (5, 5, 3, 32), Bias shape: (32,)
dtype: float32
Params: 2,432
Memory: 9728 bytes

Layer: conv2d_4 (Conv2D)
Weights shape: (3, 3, 64, 64), Bias shape: (64,)
dtype: float32
Params: 36,928
Memory: 147712 bytes
```

conv2d first 16 filters

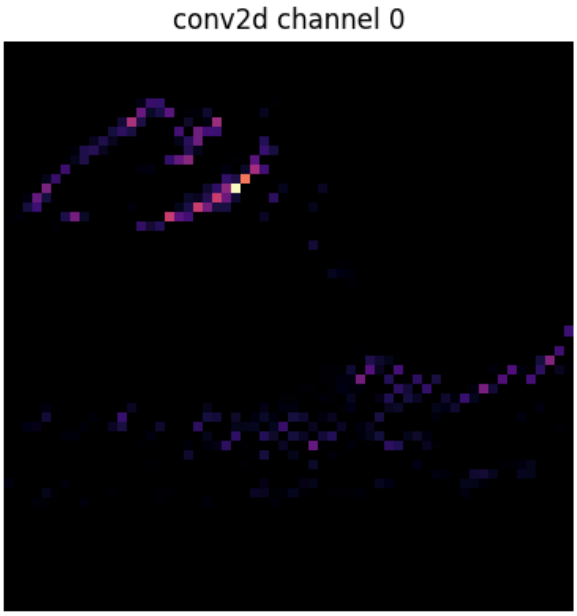


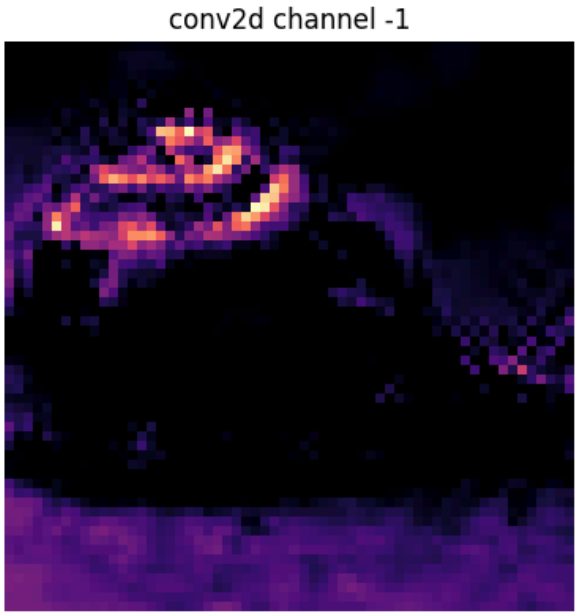


conv2d_4 first 16 filters

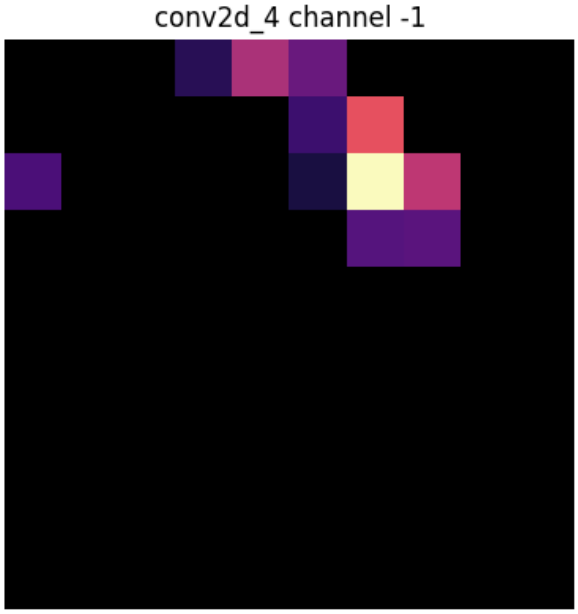
In [15]:
```python
# We can view the layer outputs as well. Here we consider them as images of
# TODO: Visualize the 2 convolutional layers outputs (intermediate feature m

# TODO: Your Code Here

example = next(iter(ds_val.batch(1)))
x = tf.cast(example["image"], tf.float32) /  255.0

acts = {}
t = x
for layer in model.layers:
    try:
        t = layer(t, training = False)
    except TypeError:
        t = layer(t)
    if layer is first_conv:
        acts["first"] = t.numpy()
    if layer is last_conv:
        acts["last"] = t.numpy()

feat_first = acts["first"]
feat_last = acts["last"]

def show_feature_maps(F, layer_name, channels=(0, -1), cmap = "magma"):
    F0 = np.asarray(F)[0]
    H, W, C = F0.shape
    print(f"Layer name: {layer_name} ")
    print(f"feature map shape = {(1, H, W, C)}")
    print(f"dtype{F.dtype}")
    print(f"Data size: {F.nbytes}")
    plt.figure(figsize=(4*len(channels), 4))
    for i, ch in enumerate(channels):
        fm = F0[..., ch]
        fm = (fm - fm.min()) / (fm.max() - fm.min() + 1e-8)
        ax = plt.subplot(1, len(channels), i+1)
        ax.imshow(fm, cmap=cmap)
        ax.set_title(f"{layer_name} channel {ch}")
        ax.axis("off")
    plt.tight_layout()
    plt.show()

show_feature_maps(feat_first, first_conv.name)
show_feature_maps(feat_last, last_conv.name)
```

```
Layer name: conv2d
feature map shape = (1, 60, 60, 32)
dtypefloat32
Data size: 460800
```






```
Layer name: conv2d_4
feature map shape = (1, 10, 10, 64)
dtypefloat32
Data size: 25600
```



In [16]:
```
# TODO: Export the filters/weights se we can use them later
# Make a directory for our image data
model_dir = os.path.abspath('model_data')
pathlib.Path(model_dir).mkdir(exist_ok=True)


# Export each image
conv_index = dense_index = 1 # layer index starts from one
for layer_idx, layer in enumerate(model.layers):
    if re.match(r'(conv|dense)', layer.name):
        weight_file_name = os.path.join(model_dir, f'{layer.name}_weights.bi
        bias_file_name = os.path.join(model_dir, f'{layer.name}_bias.bin')
    else: continue
```

```
    assert layer.weights[0].name.endswith('kernel')
    assert layer.weights[1].name.endswith('bias')

    # TODO: Your Code Here
    W, b = layer.get_weights()

    with open(weight_file_name, "wb") as wf:
        wf.write(W.tobytes())

    with open(bias_file_name, "wb") as bf:
        bf.write(b.tobytes())
```

In [17]:
```
# TODO: Export the intermediate layer outputs for each of the input for all
img_dir = os.path.abspath('img_data')
pathlib.Path(img_dir).mkdir(exist_ok=True)

for img_idx, img in enumerate(sample_imgs):
    file_dir = os.path.join(img_dir, f'test_input_{img_idx}')
    pathlib.Path(file_dir).mkdir(exist_ok=True)

    # TODO: Your Code Here
    x = tf.cast(img["image"], tf.float32)[None, ...] / 255.0
    t = x
    for layer in model.layers:
        try:
            t = layer(t, training = False)
        except TypeError:
            t = layer(t)

        arr = t.numpy()
        out_path = os.path.join(file_dir, f"{layer.name}.bin")
        with open(out_path, "wb") as f:
            f.write(arr.tobytes())
```

Tensorboard

In [18]:
```
tf.config.optimizer.set_jit(False)
tf.config.run_functions_eagerly(False)

# Setup for profiling
opts = tf.profiler.experimental.ProfilerOptions(
    host_tracer_level=2, python_tracer_level=0, device_tracer_level=1
)

log_dir = os.path.abspath(os.path.join('log_data'))
log_dir_run = os.path.abspath(os.path.join(log_dir, datetime.datetime.now().
pathlib.Path(log_dir_run).mkdir(exist_ok=True, parents=True)

try:
    tf.profiler.experimental.stop()
except:
    test = 2
```

```
finally:
    test = 1
```

In [19]:

```
# TODO: Sample Profiling - Inference for a single image:

import time


# tf.profiler.experimental.start(log_dir_run)

latency = []

@tf.function(jit_compile=False)
def run_infer(x):
    return model(x, training = False)

_ = run_infer(tf.zeros((1,64,64,3), tf.float32))

# Perform the inference profiling:
for i, image in enumerate(sample_imgs):
    # Starts Profile logging
    sub_dir = os.path.join(log_dir_run, f'img_{i}')
    x = tf.cast(image["image"], tf.float32)
    x = tf.expand_dims(x, axis = 0) / 255.0
    tf.profiler.experimental.start(sub_dir, options=opts)
    try:
        tf.profiler.experimental.Trace("single_inference", step_num = 0, _r
        t0 = time.perf_counter()
        pred = run_infer(x)
        _ = pred.numpy()
        t1 = time.perf_counter()
        latency.append((t1-t0) * 1e3)
    finally:
        tf.profiler.experimental.stop()
        print(f'Latency for Image {i}: {latency[i]}')

    file_writer = tf.summary.create_file_writer(sub_dir)
    with file_writer.as_default():
        tf.summary.scalar("latency_ms", latency[i], step=0)
    file_writer.flush()
```

```
2025-09-08 12:07:49.815893: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:07:49.815914: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
2025-09-08 12:07:49.818496: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:07:49.819597: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:07:49.822720: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/img_0/plugins/profile/2025_09_08_12_07_4
9/co2050-19.ece.iastate.edu.xplane.pb
```

```
Latency for Image 0: 2.2077089997765142
Latency for Image 1: 6.276489999436308
Latency for Image 2: 7.662865999009227
```

```
2025-09-08 12:07:50.057950: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:07:50.057982: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
2025-09-08 12:07:50.064340: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:07:50.064794: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:07:50.067260: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/img_1/plugins/profile/2025_09_08_12_07_5
0/co2050-19.ece.iastate.edu.xplane.pb
2025-09-08 12:07:50.073327: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:07:50.073365: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
2025-09-08 12:07:50.081177: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:07:50.081823: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:07:50.084451: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/img_2/plugins/profile/2025_09_08_12_07_5
0/co2050-19.ece.iastate.edu.xplane.pb
```

In [20]:
```
# Load the TensorBoard notebook extension.
%load_ext tensorboard

# Launch TensorBoard and navigate to the Profile tab to view performance pro
# *** Please note just execute this command once in a session and
# then logs for subsequent runs would be auto detected in tensorboard- url:
print(log_dir_run)
%tensorboard --logdir={log_dir_run} --port=6006

# You can view the tensorboard in the browser url: http://localhost:6006/

# Useful command line to have if tensorboard is misbehaving: kill $(ps -e |
```

```
/home/buenting/cpre_5870/lab1/Lab1/log_data/20250908-120749
```

In [21]:
```
# TODO: Sample Profiling - Online Inference:

# Vary this from 10, 100, 1000 to simulate multiple online inference
loop_index = [10, 100, 1000]

for idx in loop_index:
    sub_dir = os.path.join(log_dir_run, f'loop_{idx}')
    tf.profiler.experimental.start(sub_dir, options=opts)
```

```
    for img in ds_val.batch(1).take(idx):
        x = tf.cast(img['image'], tf.float32)/255.0
        tf.profiler.experimental.Trace("Multiple Online Inference Profiling"
        pred = run_infer(x)
        _ = pred.numpy()
    tf.profiler.experimental.stop()
    print(f'latency test for set {idx} done')
    file_writer = tf.summary.create_file_writer(sub_dir)
    with file_writer.as_default():
        tf.summary.scalar("latency_ms", latency[i], step=0)
    file_writer.flush()
```

```
2025-09-08 12:07:56.196756: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:07:56.196816: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
2025-09-08 12:07:56.288247: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
2025-09-08 12:07:56.290360: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:07:56.302570: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:07:56.305471: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/loop_10/plugins/profile/2025_09_08_12_07_
56/co2050-19.ece.iastate.edu.xplane.pb
2025-09-08 12:07:56.313743: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:07:56.313755: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
latency test for set 10 done
2025-09-08 12:07:56.639491: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
2025-09-08 12:07:56.640322: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:07:56.661568: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:07:56.663863: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/loop_100/plugins/profile/2025_09_08_12_07
_56/co2050-19.ece.iastate.edu.xplane.pb
2025-09-08 12:07:56.678646: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:07:56.678660: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
latency test for set 100 done
```

```
2025-09-08 12:08:00.356905: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
2025-09-08 12:08:00.357790: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:08:00.485052: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:08:00.488109: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/loop_1000/plugins/profile/2025_09_08_12_0
8_00/co2050-19.ece.iastate.edu.xplane.pb
latency test for set 1000 done
```

In [22]:
```python
# TODO: Sample Profiling - Batch Inference:

# We would only perform batch inference for a subset of validation set i.e.
# using different batch sizes of 20, 40, 100, 200

# Decides the size of the batch. Try: 20, 40, 100, 200
batch_size = [20, 40, 100, 200]

for batch in batch_size:
    sub_dir = os.path.join(log_dir_run, f'batch_{batch}')
    tf.profiler.experimental.start(sub_dir, options=opts)
    for imgs in ds_val.take(1000).batch(batch):
        imgs_stzd = tf.cast(imgs['image'], tf.float32)/255.0
        tf.profiler.experimental.Trace("Batch Inference Profiling", step_num
        pred = run_infer(imgs_stzd)
        _ = pred.numpy()
    tf.profiler.experimental.stop()
    print(f'latency test for batch {batch} done')

    file_writer = tf.summary.create_file_writer(sub_dir)
    with file_writer.as_default():
        tf.summary.scalar("latency_ms", latency[i], step=0)
    file_writer.flush()
```

```
2025-09-08 12:08:00.630169: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:08:00.630242: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
2025-09-08 12:08:01.245912: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
2025-09-08 12:08:01.247042: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:08:01.276035: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:08:01.278632: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/batch_20/plugins/profile/2025_09_08_12_08
_01/co2050-19.ece.iastate.edu.xplane.pb
2025-09-08 12:08:01.300430: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:08:01.300446: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
latency test for batch 20 done
```

```
2025-09-08 12:08:01.782411: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
2025-09-08 12:08:01.783737: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:08:01.810609: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:08:01.813037: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/batch_40/plugins/profile/2025_09_08_12_08
_01/co2050-19.ece.iastate.edu.xplane.pb
2025-09-08 12:08:01.834157: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:08:01.834175: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
latency test for batch 40 done
2025-09-08 12:08:02.293801: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
2025-09-08 12:08:02.294883: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:08:02.319076: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:08:02.321490: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/batch_100/plugins/profile/2025_09_08_12_0
8_02/co2050-19.ece.iastate.edu.xplane.pb
2025-09-08 12:08:02.339561: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:104] Profiler session initializing.
2025-09-08 12:08:02.339574: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:119] Profiler session started.
latency test for batch 100 done
latency test for batch 200 done
2025-09-08 12:08:02.760232: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
2025-09-08 12:08:02.761261: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:70] Profiler session collecting data.
2025-09-08 12:08:02.785159: I external/local_tsl/tsl/profiler/lib/profiler_s
ession.cc:131] Profiler session tear down.
2025-09-08 12:08:02.787648: I external/local_tsl/tsl/profiler/rpc/client/sav
e_profile.cc:144] Collecting XSpace to repository: /home/buenting/cpre_5870/
lab1/Lab1/log_data/20250908-120749/batch_200/plugins/profile/2025_09_08_12_0
8_02/co2050-19.ece.iastate.edu.xplane.pb
```

Training

In [23]:

```python
# Setup for model training
from tensorflow.keras import Model, datasets
from tensorflow.keras.models import Sequential
from tensorflow.keras.losses import categorical_crossentropy
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.layers import Dense, Flatten, Conv2D, AveragePooling2D

train_dir = os.path.abspath(os.path.join('train_data', datetime.datetime.now
pathlib.Path(train_dir).mkdir(exist_ok=True, parents=True)
```

```
# Using early stopping to monitor validation accuracy
callbacks = [
    tf.keras.callbacks.EarlyStopping(
        # Stop training when `val_loss` is no longer improving
        monitor="val_loss",
        # "no longer improving" being defined as "no better than 1e-2 less"
        min_delta=1e-2,
        # "no longer improving" being further defined as "for at least 2 epo
        patience=2,
        verbose=1,
    ),
    tf.keras.callbacks.TensorBoard(log_dir=train_dir, histogram_freq=1)
]
```

In [24]:

```
# Basic CNN model
train_model = Sequential()

# conv1
train_model.add(Conv2D(32, (5, 5), input_shape=(64, 64, 3), activation='relu
train_model.add(Conv2D(32, (5,5),activation='relu'))
train_model.add(MaxPooling2D(pool_size=(2, 2)))
train_model.add(Conv2D(64, (3,3), activation='relu'))
train_model.add(Conv2D(64, (3,3), activation='relu'))
train_model.add(MaxPooling2D(pool_size=(2, 2)))
train_model.add(Conv2D(64, (3,3), activation='relu'))
train_model.add(Conv2D(128, (3,3), activation='relu'))
train_model.add(MaxPooling2D(pool_size=(2, 2)))
train_model.add(Flatten())

# fc1
train_model.add(Dense(256, activation='relu'))

# fc2
train_model.add(Dense(200, activation='softmax'))

train_model.compile(loss='categorical_crossentropy', optimizer='adam', metri
# TODO: Consider looking at different optimizers and learning rate settings
train_model.summary()
```

```
/home/buenting/cpre_5870/lab1/Lab1/lab1_venv/lib64/python3.9/site-packages/k
eras/src/layers/convolutional/base_conv.py:113: UserWarning: Do not pass an
`input_shape`/`input_dim` argument to a layer. When using Sequential models,
prefer using an `Input(shape)` object as the first layer in the model instea
d.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

Model: "sequential"

Layer (type)	Output Shape	Par
conv2d (Conv2D)	(None, 60, 60, 32)	2
conv2d_1 (Conv2D)	(None, 56, 56, 32)	25
max_pooling2d (MaxPooling2D)	(None, 28, 28, 32)	
conv2d_2 (Conv2D)	(None, 26, 26, 64)	18
conv2d_3 (Conv2D)	(None, 24, 24, 64)	36
max_pooling2d_1 (MaxPooling2D)	(None, 12, 12, 64)	
conv2d_4 (Conv2D)	(None, 10, 10, 64)	36
conv2d_5 (Conv2D)	(None, 8, 8, 128)	73
max_pooling2d_2 (MaxPooling2D)	(None, 4, 4, 128)	
flatten (Flatten)	(None, 2048)	
dense (Dense)	(None, 256)	524
dense_1 (Dense)	(None, 200)	51

Total params: 770,216 (2.94 MB)

Trainable params: 770,216 (2.94 MB)

Non-trainable params: 0 (0.00 B)

In [ ]:
```
# TODO: Attempt to train your own model with different batch sizes
# TODO: See how long this takes without a GPU on your VDI or 2050 Coover mac
# TODO: THEN log in to your GPU VM, set ENABLE_GPU = False in the very first
# TODO: Make sure you have exported the LD_LIBRARY_PATH as the lab manual in

def normalize_img(image, label):
    return tf.cast(image, tf.float32) / 255., label

def to_categorical(image, label):
    label = tf.one_hot(tf.cast(label, tf.int32), 200)
    return tf.cast(image, tf.float32), tf.cast(label, tf.int64)

ds_re = tiny_imagenet_builder.as_dataset(as_supervised=True)
ds_retrain, ds_reval = ds_re["train"], ds_re["validation"]

ds_retrain = ds_retrain.cache().shuffle(1024)
ds_reval = ds_reval.cache().shuffle(1024)

ds_retrain = ds_retrain.map(normalize_img, num_parallel_calls=tf.data.AUTOTU
ds_reval = ds_reval.map(normalize_img, num_parallel_calls=tf.data.AUTOTUNE)

ds_retrain = ds_retrain.map(to_categorical, num_parallel_calls=tf.data.AUTOT
ds_reval = ds_reval.map(to_categorical, num_parallel_calls=tf.data.AUTOTUNE)

epoch_size = 20
```

```
init_weights = train_model.get_weights()

for batch_size in [32, 64, 128]:
    # Setup our batched datasets
    ds_retrain_batches = ds_retrain.batch(batch_size)
    ds_reval_batches = ds_reval.batch(batch_size)

    train_model.fit(
        ds_retrain_batches,
        epochs = epoch_size,
        validation_data = ds_reval_batches,
        callbacks = callbacks
    )

    train_model.save(os.path.join(train_dir, f'CNN_Train_Batch_Size_{batch_s

    total = ds_reval.cardinality().numpy()

    for images, labels in ds_reval.batch(batch_size):
        preds = train_model.predict(tf.cast(images, tf.float32)/255.0, verbo
        labels_ids = np.argmax(labels.numpy(), axis = 1)

        for i, pred in enumerate(preds):
            top_1 = np.argmax(pred)
            if top_1 == labels_ids[i]:
                acc_top1 += 1

            top_5 = tf.math.top_k(pred, k=5).indices.numpy()
            if labels_ids[i] in top_5:
                acc_top5 += 1

    print(f"Top 1 Accuracy: {acc_top1/total}\n")
    print(f"Top 5 Accuracy: {acc_top5/total}\n")

    train_model.set_weights(init_weights)
```

```
Epoch 1/20
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 129s 41ms/step - accuracy: 0.0048 - loss: 5.2
992 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 2/20
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 144s 46ms/step - accuracy: 0.0046 - loss: 5.2
992 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 3/20
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 144s 46ms/step - accuracy: 0.0046 - loss: 5.2
994 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 3: early stopping
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g. `model.save('my_model.
keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
```

```
Top 1 Accuracy: 0.248

Top 5 Accuracy: 0.519

Epoch 1/20
   1/1563 ━━━━━━━━━━━━━━━━━━━━ 4:16 164ms/step - accuracy: 0.0000e+00 - los
s: 5.2995
2025-09-08 12:15:17.750929: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
1563/1563 ━━━━━━━━━━━━━━━━━━━━ 136s 87ms/step - accuracy: 0.0048 - loss: 5.3
001 - val_accuracy: 0.0050 - val_loss: 5.2983
Epoch 2/20
1563/1563 ━━━━━━━━━━━━━━━━━━━━ 137s 88ms/step - accuracy: 0.0044 - loss: 5.2
989 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 3/20
1563/1563 ━━━━━━━━━━━━━━━━━━━━ 137s 88ms/step - accuracy: 0.0047 - loss: 5.2
991 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 3: early stopping
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g. `model.save('my_model.
keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
Top 1 Accuracy: 0.253

Top 5 Accuracy: 0.544

Epoch 1/20
2025-09-08 12:22:20.143019: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
782/782 ━━━━━━━━━━━━━━━━━━━━ 135s 172ms/step - accuracy: 0.0048 - loss: 5.29
97 - val_accuracy: 0.0050 - val_loss: 5.2983
Epoch 2/20
782/782 ━━━━━━━━━━━━━━━━━━━━ 137s 176ms/step - accuracy: 0.0044 - loss: 5.29
87 - val_accuracy: 0.0050 - val_loss: 5.2983
Epoch 3/20
782/782 ━━━━━━━━━━━━━━━━━━━━ 139s 177ms/step - accuracy: 0.0041 - loss: 5.29
88 - val_accuracy: 0.0050 - val_loss: 5.2983
Epoch 3: early stopping
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g. `model.save('my_model.
keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
Top 1 Accuracy: 0.258

Top 5 Accuracy: 0.569

2025-09-08 12:29:19.773799: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```

In [26]:

```python
# TODO: Train your model with 3 different numbers of epochs
batch_size = 32

# Setup your datasets
ds_retrain_batches = ds_retrain.batch(batch_size)
ds_reval_batches = ds_reval.batch(batch_size)
```

```
for epoch_size in [3, 10, 100]:
    train_model.fit(
        ds_retrain_batches,
        epochs = epoch_size,
        validation_data = ds_reval_batches,
        callbacks = callbacks
    )

    # Save the cnn model
    train_model.save(os.path.join(train_dir, f'CNN_Train_Epoch_Size_{epoch_s

    #Get top 1 and top 5 percentages
    for images, labels in ds_reval.batch(batch_size):
        preds = train_model.predict((tf.cast(images, tf.float32)/255.0), ver
        labels_ids = np.argmax(labels.numpy(), axis = 1)

        for i, pred in enumerate(preds):
            top_1 = np.argmax(pred)
            if top_1 == labels_ids[i]:
                acc_top1 += 1

            top_5 = tf.math.top_k(pred, k=5).indices.numpy()
            if labels_ids[i] in top_5:
                acc_top5 += 1

    print(f"Top 1 Accuracy: {acc_top1/total}\n")
    print(f"Top 5 Accuracy: {acc_top5/total}\n")
    train_model.set_weights(init_weights)
```

```
Epoch 1/3
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 142s 45ms/step - accuracy: 0.0044 - loss: 5.3
005 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 2/3
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 141s 45ms/step - accuracy: 0.0042 - loss: 5.2
992 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 3/3
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 142s 45ms/step - accuracy: 0.0044 - loss: 5.2
994 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 3: early stopping
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g. `model.save('my_model.
keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
Top 1 Accuracy: 0.263

Top 5 Accuracy: 0.594

Epoch 1/10
  3/3125 ━━━━━━━━━━━━━━━━━━━━ 2:17 44ms/step - accuracy: 0.0191 - loss: 5.3
195
2025-09-08 12:36:42.796042: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```

```
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 142s 45ms/step - accuracy: 0.0049 - loss: 5.3
001 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 2/10
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 144s 46ms/step - accuracy: 0.0046 - loss: 5.2
993 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 3/10
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 144s 46ms/step - accuracy: 0.0047 - loss: 5.2
994 - val_accuracy: 0.0050 - val_loss: 5.2985
Epoch 3: early stopping
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g. `model.save('my_model.
keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
Top 1 Accuracy: 0.268

Top 5 Accuracy: 0.619

Epoch 1/100
  3/3125 ━━━━━━━━━━━━━━━━━━━━ 2:01 39ms/step - accuracy: 0.0000e+00 - loss:
5.4552
2025-09-08 12:44:10.273935: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 142s 45ms/step - accuracy: 0.0046 - loss: 5.3
008 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 2/100
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 142s 45ms/step - accuracy: 0.0046 - loss: 5.2
993 - val_accuracy: 0.0050 - val_loss: 5.2984
Epoch 3/100
3125/3125 ━━━━━━━━━━━━━━━━━━━━ 143s 46ms/step - accuracy: 0.0044 - loss: 5.2
994 - val_accuracy: 0.0050 - val_loss: 5.2985
Epoch 3: early stopping
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g. `model.save('my_model.
keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
Top 1 Accuracy: 0.273

Top 5 Accuracy: 0.644

2025-09-08 12:51:35.180333: W tensorflow/core/framework/local_rendezvous.cc:
404] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```

Above and Beyond

In [27]:
```
# Benchmark our dataset to make sure loading our data isn't a bottleneck ...
# (This can be skipped since it can take a bit and is't all that important)

# tfds.benchmark(ds_train.batch(32), batch_size=32, num_iter=2**20)
```

In [28]:
```
# Explore new models to find a higher-accuracy model. Does the new model req
```