Convolutional Neural Networks

Project: Write an Algorithm for Landmark Classification

A simple app

In this notebook we build a very simple app that uses our exported model.

> Note how we are not importing anything from our source code (we do not use any module from the `src` directory). This is because the exported model, differently from the model weights, is a standalone serialization of our model and therefore it does not need anything else. You can ship that file to anybody, and as long as they can import `torch`, they will be able to use your model. This is very important for releasing pytorch models to production.

Test your app

Go to a search engine for images (like Google Images) and search for images of some of the landmarks, like the Eiffel Tower, the Golden Gate Bridge, Machu Picchu and so on. Save a few examples locally, then upload them to your app to see how your model behaves!

The app will show the top 5 classes that the model think are most relevant for the picture you have uploaded

In [1]:

```
import io
import torch
import torchvision.transforms as T
from PIL import Image
from ipywidgets import VBox, Button, FileUpload, Output, Label
from IPython.display import display
import matplotlib.pyplot as plt
import numpy as np

# Decide which model you want to use among the ones exported
learn_inf = torch.jit.load('checkpoints/transfer_exported.pt')

def on_click_classify(change):
    # Load image that has been uploaded
    img = Image.open(io.BytesIO(btn_upload.value[-1].content))
    img.load()

    # Let's clear the previous output (if any)
    out_pl.clear_output()

    # Display the image
    with out_pl:
        ratio = img.size[0] / img.size[1]
        c = img.copy()
        c.thumbnail([ratio * 200, 200])
        display(c)

    # Transform to tensor
    timg = T.ToTensor()(img).unsqueeze_(0)

    # Calling the model
    softmax = learn_inf(timg).data.cpu().numpy().squeeze()

    # Get the indexes of the classes ordered by softmax (larger first)
    idxs = np.argsort(softmax)[::-1]

    # Loop over the classes with the largest softmax
    for i in range(5):
        # Get softmax value and class name
        p = softmax[idxs[i]]
        landmark_name = learn_inf.class_names[idxs[i]]
        labels[i].value = f"{landmark_name} (prob: {p:.2f})"

# Initialize widgets
btn_upload = FileUpload()
btn_run = Button(description="Classify")
out_pl = Output()
labels = [Label() for _ in range(5)]

# Attach the classification function to the 'classify' button
btn_run.on_click(on_click_classify)

# Organize widgets in layout
vbox_layout = VBox([Label("Please upload a picture of a landmark"), btn_upload, btn_run, out_pl] + labels)

# Display the GUI
display(vbox_layout)
```

Please upload a picture of a landmark

Upload (1)

Classify

09.Golden_Gate_Bridge (prob: 0.80)

38.Forth_Bridge (prob: 0.10)

30.Brooklyn_Bridge (prob: 0.10)

28.Sydney_Harbour_Bridge (prob: 0.01)

33.Sydney_Opera_House (prob: 0.00)

(optional) Standalone app or web app

You can run this notebook as a standalone app on your computer by following these steps:

1. Download this notebook in a directory on your machine
2. Download the model export (for example, `checkpoints/transfer_exported.pt`) in a subdirectory called `checkpoints` within the directory where you save the app.ipynb notebook
3. Install voila if you don't have it already (`pip install voila`)
4. Run your app: `voila app.ipynb --show_tracebacks=True`
5. Customize your notebook to make your app prettier and rerun voila

You can also deploy this app as a website using Binder: https://voila.readthedocs.io/en/stable/deploy.html#deployment-on-binder

Create your submission archive

Now that you are done with your project, please run the following cell. It will generate a file containing all the code you have written, as well as the notebooks. Please submit that file to complete your project

In [3]:
```
!python src/create_submit_pkg.py
```

```
executing: jupyter nbconvert --to html app.ipynb
[NbConvertApp] Converting notebook app.ipynb to html
[NbConvertApp] Writing 588864 bytes to app.html
[NbConvertApp] Converting notebook cnn_from_scratch.ipynb to html
[NbConvertApp] Writing 2151556 bytes to cnn_from_scratch.html
[NbConvertApp] Converting notebook transfer_learning.ipynb to html
[NbConvertApp] Writing 1701012 bytes to transfer_learning.html


executing: jupyter nbconvert --to html cnn_from_scratch.ipynb
executing: jupyter nbconvert --to html transfer_learning.ipynb
Adding files to submission_2024-01-14T18h11m.tar.gz
src\create_submit_pkg.py
src\data.py
src\helpers.py
src\model.py
src\optimization.py
src\predictor.py
src\train.py
src\transfer.py
src\__init__.py
app.ipynb
cnn_from_scratch.ipynb
transfer_learning.ipynb
app.html
cnn_from_scratch.html
transfer_learning.html

----------------------------------------------------------------
Done. Please submit the file submission_2024-01-14T18h11m.tar.gz
----------------------------------------------------------------
```

In [ ]:
```
```