Deep learning for audio and music

Lab 2 – convolutional neural networks for audio

Part 1

For the least bittern `coo coo coo' breeding call:

- 0.5 seconds long, 500-600 Hz
- Recorded at 44.1 kHz
- Spectrogram: frame size 1024, 512 hop

Calculate: bandwidth of a single spectrogram bin; spectrogram frame rate; dimensions of the "rectangle" containing the call.

For a simple network:

3x3 conv → 2x2 maxpool → 3x3 conv

Calculate the receptive field of a pixel in the output layer.
Does it accommodate the least bittern call?

Alternatively, try it using the settings as used in the "bulbul" bird detection network (Grill & Schlueter 2016):

- Recordings downsampled to 22.05 kHz
- Spectrogram: frame size 1024, hop size 315 samples

Network architecture:

3x3 conv → 3x3 maxpool →
3x3 conv → 3x3 maxpool →
3x**1** conv → 3x**1** maxpool →
3x**1** conv → 3x**1** maxpool

Does the receptive field accommodate the least bittern call?

Part 2

Using PyTorch, implement and train a simple CNN as follows:

- **Data:** warblrb2k (note that this dataset is for a binary classification problem)
- **Input features:** spectrograms – feel free to reuse spectrogram code, e.g. from previous lab.
- **Network architecture:** use the simpler architecture above, and add two dense layers at the end

You can follow the standard PyTorch CNN tutorial for most of this. You will need to write your own code to iterate through our dataset and pass spectrograms to the network.