Urban Sound Classification with Neural Networks

Use TensorFlow 2 to train neural networks for the classification of sound events based on audio files from the UrbanSound8K dataset: it contains 8732 sound excerpts (<=4 seconds) of urban sounds labeled with 10 classes: air_conditioner, car_horn, children_playing, dog_bark, drilling, enginge_idling, gun_shot, jackhammer, siren, and street_music. Further information about the dataset is available in:
J. Salamon, C. Jacoby and J. P. Bello, "A Dataset and Taxonomy for Urban Sound Research", 22nd ACM International Conference on Multimedia, Orlando USA, Nov. 2014. (available here)
A sound file can be read using the Python librosa library (doc) as follows:

```
file_path = './UrbanSound8K/audio/fold4/7389-1-4-14.wav'
sound_file, sampling_rate = librosa.load(file_path, sr = None)
```

The sound_file is returned as a numpy array where each entry is a discretized sample, with sampling_rate of 44100Hz (which is the same for all the traces of the dataset). The length in seconds of all the audio traces is approximately 4 seconds. The files metadata are available in the UrbanSound8K.csv file provided with the dataset which include, for instance, the class for each audio trace. Use the classes in the UrbanSound8K.csv as labels to predict.
Neural networks can be fed with raw data or proper features can be extracted. The librosa library can be used to extract several useful features common in the field of sound events classification. For instance, the Mel-frequency cepstral coefficients (MFCCs) can be extracted as follows:

```
mfcc_coefficients = librosa.feature.mfcc(y = sound_file, sr = sampling_rate, n_mfcc = 50)
```

Other features can be extracted with the librosa library (or even other libraries) including the Chromagram (with the librosa.feature.chroma_stft). Other less tailored features for the considered domain can be extracted, e.g. the Root Mean Square of the audio signal and its statistics (min, max, average, standard deviation, etc.). Feature selection methods may be applied to select most relevant features (for instance PCA. A wide list of methods is implemented in sklearn.feature_selection).
The dataset is provided with 10 predefined folds. Train the model on folds: 1, 2, 3, 4, 6, and test it on folds: 5, 7, 8, 9, 10. Report the obtained average accuracy and standard deviation across the test folds. Experiment with different feature extraction methods, network architectures and training parameters documenting their influence of the final predictive performance.

If you are not familiar with neural networks for image classification, take one of the many tutorials available in TensorFlow.