**DEEP LEARNING PROJECT PLAN SUMMARY**

* **Research objective / context and motivation**

The main idea of the project will be about sound classification but especially sound coming from urban areas. What if we can detect gun shots sounds (or any alarming sound) and differentiate them from common sounds in a split of a second. This application could help the city administration/police to notice dangerous events faster than ever.

* **Methodology**

The general idea is to transform sound data into images through feature engineer technique such as “Mel” spectrogram (i.e. scale base on pitch) or chromagram (i.e. scale base on pitch categories). This manipulation will allow us to apply a convolutional Neural Networks (CNNs) and use each feature engineer as a channel to end up with a feature maps.

* **Dataset**
  + The dataset we will use is an Urban Sound Classification dataset. This dataset contains 8 000 samples with 10 classes. It will allow us to perform our training for our model with a dataset that will contain data related to our application and meet a threshold for accuracy allowing for implementation:  
    <https://urbansounddataset.weebly.com/>
  + A secondary dataset we can use later to test our classifier better is AudioSet which contains 2 084 320 samples with 632 classes which would be to computationally heavy as an initial dataset. We can use the same 10 classes as the urban sound classification dataset to compare models later in the project. <https://research.google.com/audioset/?fbclid=IwAR3If9WF29_QwarlvzjwylQVYxxTKNhCAcpA0vanD_hhOe0e8XVfVcyFMYs>
* **Data preprocessing and manipulation**
  + **Need to do some literature review in order to tackle that feature engineering. Could we use a model that already extracts some of these features automatically.**
  + When it comes to the data preprocessing, there are many different articles on how to transform sound files directly into a spectrogram which will be the initial step for us[[1]](#footnote-1). There are also multiple libraries that exist to convert a *.WAV* file to a spectrogram representation which we plan to use[[2]](#footnote-2).
  + There is a possibility for us to use data augmentation to make sure our dataset is bigger. Not only will make it easier to train with more example, but depending on how we decide to do this it could also account for distortion in the sound received through the means we have, thus making the model more robust.
    - The following paper[[3]](#footnote-3) presents how we can use methods for data augmentation both on the audio file directly and on the spectrograms created after. We plan on using some of those methods to explore the potential improvements to the model’s accuracy.
  + To find specific methods for data augmentation on audio files we can use similar methods found in this paper. We want to use methods like, time stretching, pitch shifting and more importantly background noise from the MUDA library. Moreover, we can also create multiple sets of data so that we can figure out what the optimal data augmentation type or hybrid is[[4]](#footnote-4).
* **Estimation method and models**
  + Leverage some of the feature extraction or feature engineering by using pre-trained CNN model such as the VGG-16 model and then train a fully-connected network.
  + Based on the graph we can see in this paper we can see that an architecture we can decide to use is Convolutional Block Attention Module (CBAM) which could allow us to use attention in the estimation. This paper is also very recent[[5]](#footnote-5).
  + When it comes to a good architecture for models there is this paper[[6]](#footnote-6) here that gives us an overview of what they use for acoustic data. The type of data is slightly different, but the architecture could still help us. For their state-of—the-art they use Bag of Audio Words (BoAW) with a Deep Neural Neural (DNN) and Hidden Markov Models (HMM). Additionally, their optimal architecture is two conv layer followed by 1 pooling 3 times and 3 FC layers after.
  + There is also this paper that goes through most of the ones we know and love like ResNet [[7]](#footnote-7). Can help us at least get an idea of how we should look for them.
  + There is also a possibility to add two type of DNN such as the Recurrent Neural Network (RNN) with CNN to create a CRNN[[8]](#footnote-8). This type of architecture is relevant when we are doing temporal classification of sounds events. This could be presented as a future improvement in order to put the sound classifier in production.
* **Hyperparameters tuning**
  + Recommendation about the type of hyperparameter we should focus on vs the state-of-art default parameter.
* **Performance measure**
  + In a similar way to which one would do sentiment analysis we can look at accuracy in a similar way. Additionally, to compare architecture a simple classification error could be provided by a confusion matrix of 10x10 which would allow us to understand what the problematic classes are.
* **Questions**

1. What is the best architecture for sound classification?
2. Is the transformation into images something we should consider doing?
3. Are there any other methods known?
4. Do you see any challenges with our project?
5. What would you like to hear in the final report regarding our project?

1. <https://fairyonice.github.io/implement-the-spectrogram-from-scratch-in-python.html> [↑](#footnote-ref-1)
2. https://stackoverflow.com/questions/44787437/how-to-convert-a-wav-file-to-a-spectrogram-in-python3 [↑](#footnote-ref-2)
3. <https://arxiv.org/ftp/arxiv/papers/1912/1912.05472.pdf> [↑](#footnote-ref-3)
4. <https://arxiv.org/pdf/1608.04363.pdf> [↑](#footnote-ref-4)
5. <https://arxiv.org/ftp/arxiv/papers/1901/1901.06032.pdf> [↑](#footnote-ref-5)
6. <https://arxiv.org/pdf/1604.07160.pdf> [↑](#footnote-ref-6)
7. <https://arxiv.org/pdf/1609.09430.pdf> [↑](#footnote-ref-7)
8. <https://tutcris.tut.fi/portal/files/13594874/1702.06286.pdf> [↑](#footnote-ref-8)