Stage Memo and Presentations

Done by: Ranim Tom

Table of Contents

Learning audio representations with self-supervision [1]

Goal of this paper:

- Apply self supervision to learn general purpose audio representations using 2 self supervised task:
  - Audio2Vec: aims to reconstruct a spectrogram slice from past and future slice
  - TemporalGap: estimates the distance between two short audio segments extracted at random from the same audio clip

Introduction I

- Disadvantage of supervised learning altough the major advancement it has done:
  1. requires collecting large annotated datasets specific to each task to be solved
  2. separate models are typically trained for each task, making it difficult to reuse computational resources when multiple such models are deployed on a mobile device
- Solution to this: using unsuperivsed learning approach, specifically self supervised where:
  - formulates an auxiliary task based on the available unlabelled data
  - the model will learn some general purpose representations in a lower dimensional embedding space while solving this auxiliary task

Introduction II

  - As a result, the embedding encoder, e.g., the portion of the model architecture mapping the input data to the embedding space, can be reused as a feature extractor for different downstream tasks.
- One of the earliest successes of self-supervised learning was obtained in the context of language models is Word2Vec, where it comes in 2 format:
  1. Continuous bag-of-words (CBoW): the model predicts the current word based on the context of surrounding words
  2. skip-gram: predicts surrounding words given the current word

Self Supervised Part: Auxiliary Task Proposed

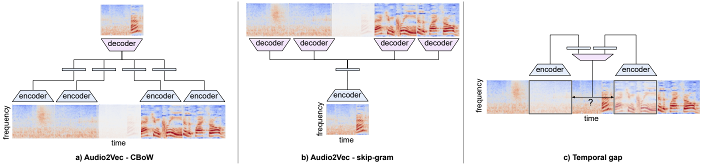


Figure 1: Overview of the proposed self-supervised learning tasks [1]

- In CBoW formulation the auxiliary task consists of reconstructing a temporal slice of pre-determined duration from a number of past and future slices.

Description of Audio2Vec: CBoW Formulation I

- Let x be an audio clip of n samples, $x = [x_1, x_2, \cdots, x_n]$
- $x \mapsto X \in \mathbb{R}^{T \times F}$, where $\begin{cases} X : \text{Sepctrogram of } x \\ T : \text{number of temporal frames} \\ F : \text{frequency bins} \end{cases}$.
- The authors randomly pick a slice of X denoted by $X_i \in \mathbb{R}^{N \times F}$ such that $N < T$.
- Let's say that $X_i = X_{(0)}$. The next step is to pick some other slices of the same audio clip x, before and after the slice $X_{(0)}$.
  - More specifically, we have past slices denoted by $\{X_{(-P)}, \cdots, X_{(-1)}\}$ and future slices denoted by $\{X_{(1)}, \cdots, X_{(P)}\}$
- Each slice from $X_{(-P)} \to X_{(P)}$ will be processed by the same encoder denoted by *Enc*, and as an output we obtain an embedding vector $z_{(p)}$.

Description of Audio2Vec: CBoW Formulation II

- $Enc(X_P) = z_{(p)}$
- Then a vector $\bar{z}_{(0)}$ is obtained by concatenating all the embedding vector and inputted to a convolutional decoder to obtain a reconstruction of the slice $X_{(0)}$
  - Concatenating $\bar{z}_{(0)} = \left[z_{(-P)}, \cdots, z_{(-1)}, z_{(1)}, \cdots, z_{(P)} \right]$
  - Decoder: $\hat{X}_{(0)} = Dec(\bar{z}_{(0)})$
- Objective Function: the overall encoder-decoder is trained using MSE: $\| X_{(0)} - \hat{X}_{(0)} \|$

Description of Audio2Vec: Encoder-Decoder Architecture I

- Concerning the decoder part:
  - Reversing the order of the convolutional layer in the encoder part
  - Replacing max-pooling with nearest-neighbor upsampling
- Raw Audio Signal Processing:
  - sampling frequency = 16 kHz
  - Window size of 25 ms and hop length of 10 ms to compute the STFT
  - Computing mel spectrogram by using 64 frequency bins in the range of 60-7800 Hz
- Encoder Architecture: 6 layers all having 3×3 filter, with channel numbers respectively equal to [64, 128, 256, 256, 512, 512]

Description of Audio2Vec: Adapting to SpecCense

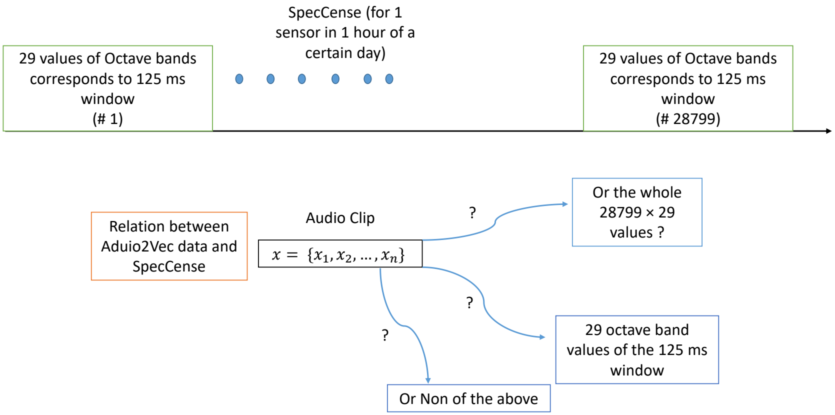
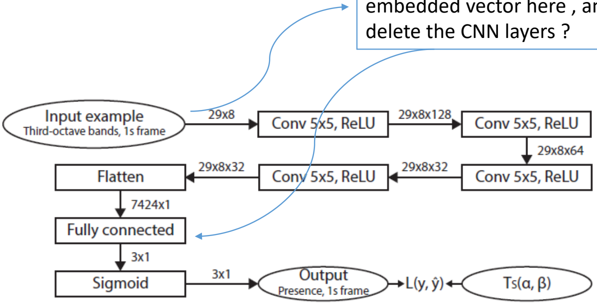


Figure 2: Relation to SpecCense Data

Description of Audio2Vec: Adapting to SpecCense

Adaptation to Felix Architecture

1st choice: Do I put the embedded vector here , and delete the CNN layers ?



Step 1: Slicing the frames

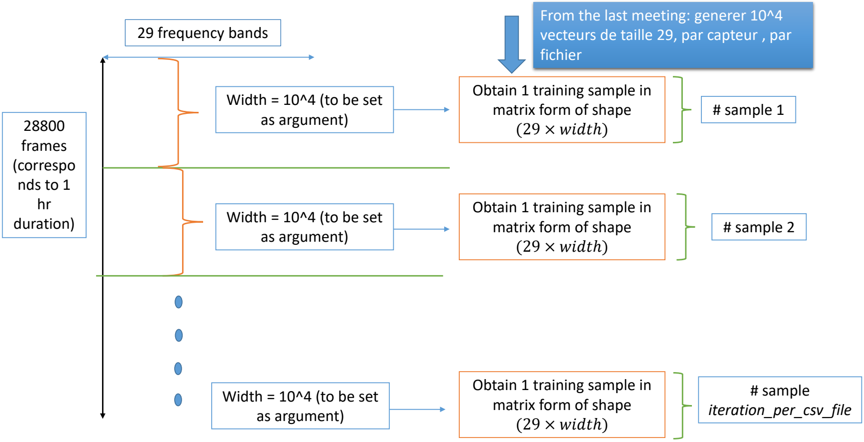
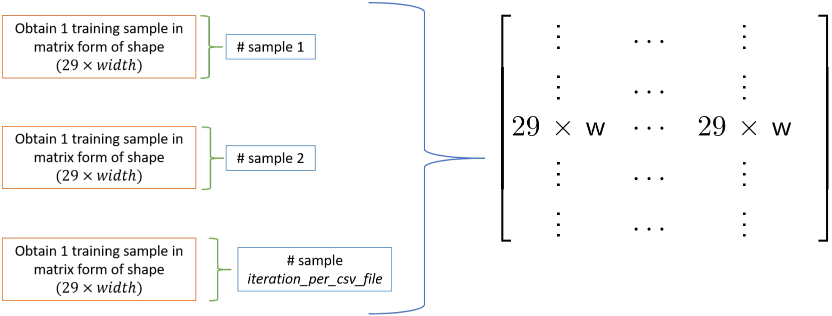


Figure 4: Frame Slicing

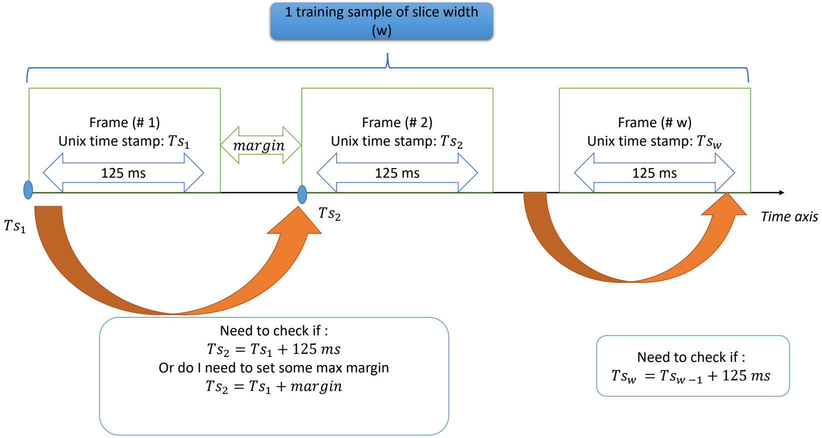
Step 2: Stacking into matrix X



- The matrix X will contain the total samples, in which we will split them into train, validation and test set.
- shape of X:

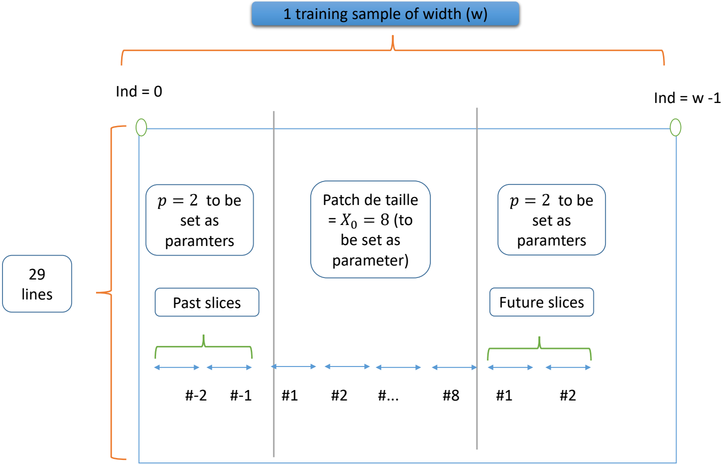
$$\begin{cases} \text{number of rows: } 29 \times \text{width} \\ \text{number of columns: iteration per csv file} \\ \text{iteration per csv file} = \text{math.floor(number of frames / width)} \end{cases}$$

Time Continuity



- In case there is a discontinuity, do I reject all the slice (the training sample)?

Pretext Task



References I

[1] M. Tagliasacchi, B. Gfeller, F. D. C. Quitry, and D. Roblek, "Learning audio representations with self-supervision," *IEEE Signal Processing Letters*, 2020.