LORRAINE UNIVERSITY

Self-Supervised Learning for Automatic Speech Recognition

Author:
Nasser-Eddine MONIR,
Zakaria BERBARA

Supervisor:
Dr Imran SHEIKH

Reviewer:
Md Sahidullah

MSc Natural Language Processing M1 Supervised Project
Realization Report

June 20, 2021

Acknowledgements

We would like to express our heartfelt gratitude to Dr. Imran S. for guiding and supporting us during the whole process of producing this final report. His willingness to share his knowledge, his dedication and enthusiasm for this project, and his comments have all been crucial.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this report we present the results obtained during the experimentation of the phonemes segmentation technic in an unsupervised learning fashion, we investigate this specific part of the Automatic Speech Recognition (ASR) guided by the method describe in the blind phoneme segmentation [1].

In ASR, Self-Supervised Learning is proposed for exploiting unlabeled data to over pass the disadvantages encountered with supervised learning. Generating a dataset with great enough annotation is expensive, while unlabeled audio data is constantly generated, Self-Supervised Learning is motivated by the desire to make use of a huge volume of unlabeled data. Self-Supervised Learning's basic premise is to create labels from unlabeled data. According to the structure or characteristics and properties of the data itself, and then train a model on this unsupervised data in a supervised manner. self-supervised learning is often used to train a model to learn the latent characteristics of the input. Computer vision, video processing, and robot control are all examples of where this technology is used.

The continuous, time-dependent character of the signal, as opposed to text processing, is one of the most challenging aspects of speech processing. As a result, several speech recognition tasks need the pre-segmentation of the voice signal into words or sub-words components such as phonemes, syllables, or words. The phoneme segmentation consists of finding boundaries which marks the transition from one phoneme to another.

So far, several potential techniques to solving this challenge have been offered. DBSLTMs for Phoneme Boundary Detection [2]. Gated Recurrent Neural Networks and Its Correlation with Phoneme Boundaries [3]. all these technics have in common the use of RNN to manage the temporal nature of the audio signal.

Our efforts are geared toward the use of LSTM to perform a next frame prediction and peak tracking over the error curve, because of the positive correlation between those brut peaks and the presence of the potential boundary. Our efforts were geared according to the hypothesis mentioned in the works cited in the original document of blind phoneme segmentation [1].

The rest of this document is organized in the following manner. In section 2 a brief presentation of the tools and major key concepts used in such a task. In section 3 more details about the audio corpus and the data preparation along with a clear presentation of the model and its hyper parameters, this section will also contain the results obtains by the end of the experimentation phase.

Chapter 2

Methodology

2.1 Feature extraction and MFCC

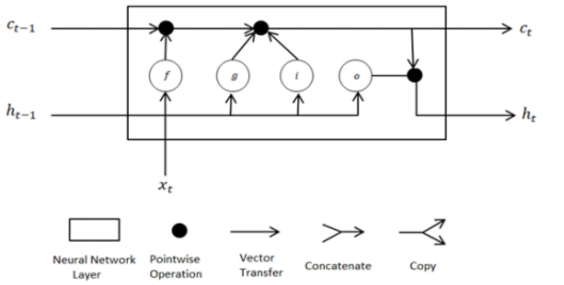
The starting point of the project is the input presentation. We perform an MFCC extraction on each speech audio, the speech signal is sliced in frames (vectors) that represents approximately a duration of 22.5 ms of the temporal audio signal, the frames are stacked in a matrix, whose dimension depends on the duration of the audio fille, and several other parameters like the framing window and the hop length, however the number of column which is the number MFCC's extracted is fixed to 13 all along our experiments.

2.2 RNNs

Recurrent Neural Networks (RNN) are a type of supervised Deep Learning architecture that works best with time series. Hidden layers are interconnected throughout time in this design, allowing them to maintain memory states from preceding layers. The network gains state or memory because of the recurrent connections, which allows it to learn and exploit the ordered character of observations. The network gains state or memory because of the recurrent connections, which allows it to learn and exploit the ordered character of observations [4]. However, when working with extended data sequences, models can run into the issue of vanishing gradients. As a result, the network is unable to function properly. To address the vanishing gradient problem, The Long Short-Term Memory (LSTM) network has been introduced as a new type of RNN [5]. Researchers have improved it significantly since then, and it is now the most often used architecture for prediction problems.

RNNs come in a variety of forms, including LSTMs. There are four layers in this architecture, each of which interacts differently: information gate, forget gate, input and output gate. The configuration of a simple LSTM memory block is shown in Figure 2.1. It should be noted that in real-world configurations, there may be more gates.

FIGURE 2.1: A simple LSTM block memory



Each component of the memory block is described in the bellows equations (1) to (6).

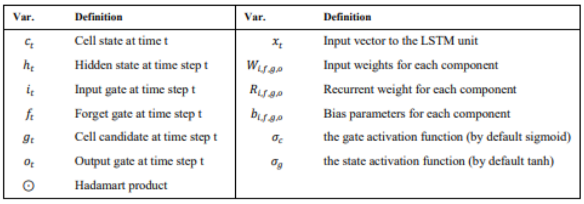
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (1) \qquad f_t = \sigma_g(W_f x_t + R_f h_{t-1} + b_f) \quad (4)$$

$$h_t = o_t \odot \sigma_c(c_t) \quad (2) \qquad g_t = \sigma_c(W_g x_t + R_g h_{t-1} + b_g) \quad (5)$$

$$i_t = \sigma_g(W_i x_t + R_i h_{t-1} + b_i) \quad (3) \qquad o_t = \sigma_g(W_o x_t + R_o h_{t-1} + b_o) \quad (6)$$

Table 2.1 shows the variable descriptions.

TABLE 2.1: Definition of variables used in equations (1) to (6)



Var.	Definition	Var.	Definition
c_t	Cell state at time t	x_t	Input vector to the LSTM unit
h_t	Hidden state at time step t	$W_{i,f,g,o}$	Input weights for each component
i_t	Input gate at time step t	$R_{i,f,g,o}$	Recurrent weight for each component
f_t	Forget gate at time step t	$b_{i,f,g,o}$	Bias parameters for each component
g_t	Cell candidate at time step t	σ_c	the gate activation function (by default sigmoid)
o_t	Output gate at time step t	σ_g	the state activation function (by default tanh)
$\odot$	Hadamart product		

2.3 Blind phoneme segmentation using RNNs

We use LSTM to perform the next frame predication on our audio training data, we still have an unsupervised training even if we keep the classic training mode of the LSTM which consists of predicting X knowing Y, with one exception is that this label Y is extracted from the raw audio data itself. The input consists of a single frame X_t (vector) and the corresponding label $Y = X_{(t+1)}$, the output is a frame of the same size of X_t (13 dimension).

Using prediction error as a segmentation criterion, the idea is to inspect the error curve of the training of the LSTM to detect the presence of a boundary, depending on a specific threshold which must be dynamically set. Then each error peak that overpass the threshold on predicting a certain frame, will be automatically tagged with a boundary.

Chapter 3

Experiments and Results

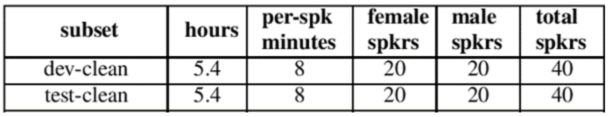
3.1 Corpus

The LibriSpeech Corpus was used in our investigation and experiments [6], it is a collection of approximately 1,000 hours of audiobooks that are a part of the LibriVox project. The Librispeech is divided into subsets, in our experiments we use the Dev clean and test clean respectively to build the training data and the validation, test data. For Both subsets, the sample rate is 16 kHz, The mean, minimal, and maximal duration of the audios are respectively 7.42, 1.29, 34.96 seconds. the table below contains furthermore details.

3.2 Pre-processing and MFCC extraction

In addition to the Librispeech audio, we use the word and phoneme alignment Librispeech that corresponds to the text transcription contained in a

TABLE 3.1: dev-clean test-clean description



subset	hours	per-spk minutes	female spkrs	male spkrs	total spkrs
dev-clean	5.4	8	20	20	40
test-clean	5.4	8	20	20	40

TextGrid format of the Dev clean and the Test clean audio files. The most important information in this TextGrid files is the part concerning the phoneme alignment (the duration of each phoneme), using the librispeech audio and text alignment, we extracted the MFCC for each audio file using the librosa python library, after obtaining the matrix representation of the audio file, we read the TextGrid and we tagged each frame with the corresponding phoneme and mark the presence or not of a boundary(where 1 and 0 corresponds respectively to the presence and the absence of a boundary).

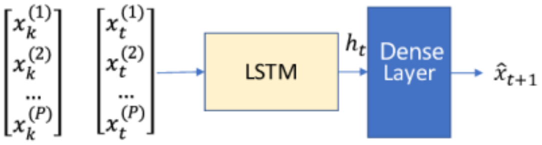
At the end of the pre-processing step, we build a training set with 2703 audios, a validation set with 787 audios, and a test set with 1833 audios.

3.3 Model architecture and hyper parameters

Our model is composed with a first LSTM layer containing 32 LSTM cells connected with a dense layer (linear layer), the input of the LSTM is of the size13, the input of the dense layer is connected with each hidden state of an LSTM unit(cell), and that why the input size of the linear layer is of the same size as the number of LSTM cells(32), finally the output of the dense layer and by consequence the output of our model is of the size 13.

The LSTM layer captures the deep temporal dependencies within the multivariate time series to comprehend the internal representation of incoming data. The dense layer is in charge of predicting future frame values.

FIGURE 3.1: Model architecture


$[x_k^{(1)}, x_k^{(2)}, \dots, x_k^{(P)}]$
$[x_t^{(1)}, x_t^{(2)}, \dots, x_t^{(P)}]$
LSTM
h_t
Dense Layer
$\hat{x}_{t+1}$

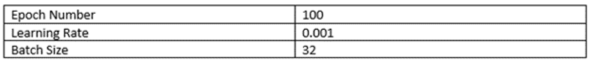
The goal of the training is to optimize the model by minimizing the root mean square error between the actual MFCC feature vector x_t and the predicted MFCC feature vector $\hat{x}_t$, the equation (7) describe the loss function.

$$\text{Loss}_{\text{(RMSE)}}(t) = \frac{||x_t - \hat{x}_t||^2}{d} \qquad (7)$$

To perform an audio segmentation - boundaries detection - a time series forecasting task is needed. In other words, given an audio frame represented by 13 MFCCs, the purpose is to predict at each iteration the next frame. Thus, we define an LSTM model whose forward pass outputs a prediction, a cell state and a hidden state for each 32x2x43 frames. An MSE loss is computed afterward to complete the backward propagation that requires the gradients to be computed in order to update the weights given the Stochastic Gradient Descent optimization problem. Besides, an evaluation part is performed on the validation set so as to avoid overfitting by using an early stopping condition. The latter consists in breaking the training once the loss stops decreasing with a delay of 3 epochs. This whole process was carried out over 100 epochs.

All the chosen hyper parameters are summarized in the table below:

TABLE 3.2: hyper parameters



Epoch Number	100
Learning Rate	0.001
Batch Size	32

3.4 Evaluation measures

precision (P), recall (R) and F1-score defined as the harmonic mean of precision and recall, is used to evaluate our system's performance on the test set containing 1833 audios. Other metrics have been created to address this problem. The R-value is one such example. More information can be found in Räsänen et al(2009)[7].The R-val is described by the equation (8):

$$\text{R-val} = 1 - \frac{\sqrt{(1-\text{R})^2 + \text{OS}^2} + \left|\frac{\text{R}+1-\text{OS}}{\sqrt{2}}\right|}{2} \qquad (8)$$

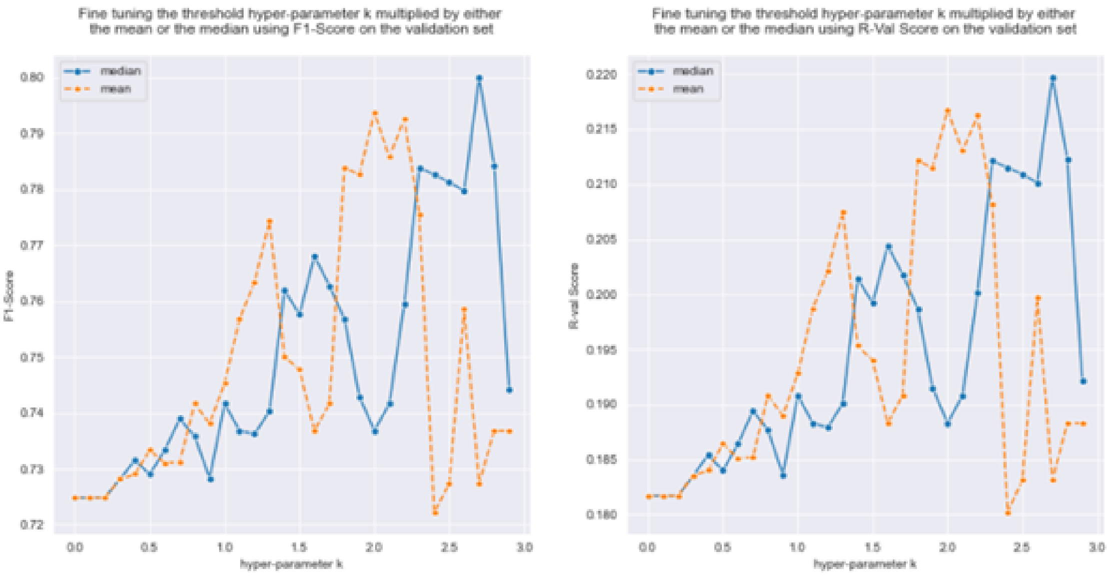
Where $os = \frac{R}{P} - 1$ is the over-segmentation measure. The R value indicates how close the segmentation is to the ideal 0 OS segmentation. 1 R point and the P=1 line in the R, OS space. the objective of our project is to spot the boundaries with in speech utterances, and this with a sufficiently satisfactory precision, because to identify boundaries at the exact time transition is something really hard to achieve. For the reason, we decided to establish a common condition to consider a boundary to be correctly detected, and it is an overlapping tolerance windows(tolerance interval) of 5 ms distance on either side.

3.5 Results and discussion

Among the 2703 speech utterances of the training set, and among 70% of test clean Librispeech utterances which represents a total of 1833 audios for testing, we achieved a 81.81% F1 score and 22.78% R-val score.

In addition to the hyper parameters mentioned in the section 3.3, We tuned an hyper parameter K that determined the value of the threshold that is crucial part of the evaluation process.The threshold is obtained by multiplying the mean or the median of the errors of an audio with K, we test the value on a range between 0 and 3, the figure 3.2 shows the result of the experimentation:

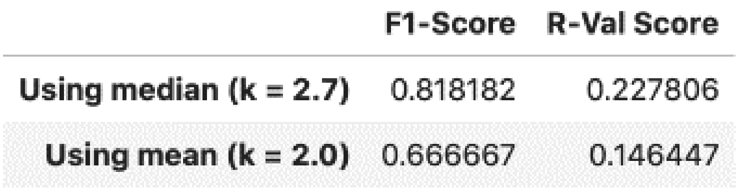
FIGURE 3.2: hyper parameter K


Fine tuning the threshold hyper-parameter k multiplied by either the mean or the median using F1-Score on the validation set

Fine tuning the threshold hyper-parameter k multiplied by either the mean or the median using R-Val Score on the validation set

We concluded that the best value of K is 2.7 and we combine it with the median.

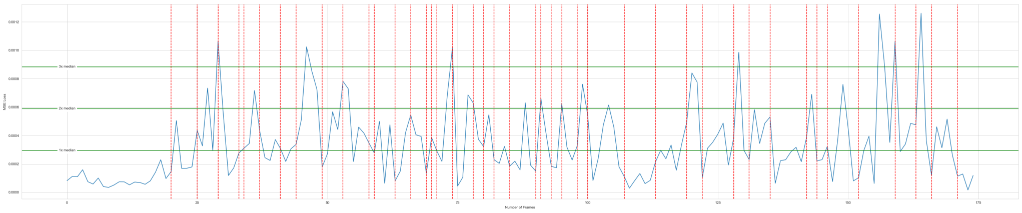
TABLE 3.3: best K value for the median and the mean of the errors



	F1-Score	R-Val Score
Using median (k = 2.7)	0.818182	0.227806
Using mean (k = 2.0)	0.666667	0.146447

The figure below display the gold boundaries(in red) for a single audio, along with error curve(in blue), and in red we have respectively 1, 2 and 3 as values for K.

FIGURE 3.3: Error curve with gold boundaries



The result is not very satisfying especially concerning the R-val, and this probably due to the very short tolerance interval 5ms, in comparison with the original paper where the authors prefers to set it on 20 ms.

3.6 Conclusion

This experiment on phoneme detection in speech utterances has been extremely eventful, to the point where it change completely our vision for the future projects. From how to approach a study subject to how to build teamwork around it, we've learned a lot.

To begin with, we've seen how a minor lapse in time management may disrupt job flow. We wasted a lot of time on activities we thought were important but were not. We will be more discriminating in future work about what is vital and what is not in order to achieve the project's ultimate goal.

Second, we had never worked on a research project that lasted many months before. It made us think big because the project could be tackled from whatever perspective we wanted. It also allowed us to concentrate our research on a single issue over a lengthy period of time, allowing us to gain a better grasp of the field of voice recognition.

Despite the fact that the results were not what we had hoped for, we are humbled to have completed such a significant endeavor.

Bibliography

[1]Blind phoneme segmentation with temporal prediction errors https://arxiv.org/pdf/1608.00508.pdf.

[2]Phoneme Boundary Detection using Deep Bidirectional LSTMs https://isl.anthropomatik.kit.edu/pdf/Franke2016.pdf.

[3] Gate Activation Signal Analysis for Gated Recurrent Neural Networks and Its Correlation with Phoneme Boundaries https://arxiv.org/pdf/1703.07588.pdf.

[4] Brownlee Jason. "Deep learning for time series forecasting." Vermont, Australia:Machine Learning Mastery, 2018.

[5] Sepp ochreiter and urgen Schmidhu er. “Long short-term memory”. Neural computation, 9(8):1735–1780, 1997.

[6] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. LibriSpeech: An ASR corpus based on public domain audio books. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5206– 5210. IEEE, 2015.

[7] Okko Räsänen, Unto Kalervo Laine, and Toomas Al- ¨ tosaar. 2009. An improved speech segmentation quality measure: the r-value. In Proceedings of Interspeech.