Technical report: voice activity detection in constrained environment

Adrien Gresse

Foreword

The system developed here is not intended for performances, even if good metrics are presented in this report. There is no doubt that this system would perform poorly when dealing with "in the wild" data. As it was not intended to be a best performing system and because we do not have great computation resources at our disposal, this work focus more on the description of the method and gives details on further improvements that could be carried out.

1 Introduction

Voice activity detection systems are an essential component of ASR. It is important to allow the detection of speech in order to separate speech segments from unvoiced segments that are considered as noise. Also, we can easily admit that voice activity detection is of great interest for autonomous devices, since it is not possible to record sound continuously. Voice activity detection, when associated with Keyword Spotting, allows autonomous devices to awake just on time. Today, most of autonomous devices have constrained memory/computation resources. In this work, we present a deep neural network based approach for voice activity detection that does not necessitate much memory nor computation capacity.

Deep learning have successfully been used for the voice activity detection task in different papers [1], [2] [3]. The authors have demonstrated the good ability for convolutionnal neural network to model acoustic information as for recurrent neural network to capture temporal information from audio sequences. In more recent works [4] the authors used both by combining trainable filters and recurrent layers to address this task. They use a particular convolutional architecture, called syncnet, as trainable filters. In this work we choose to rely on a simpler architecture. We suppose that a convolutional architecture would be able to encode relevant features with the help of temporal information coming from a RNN block. The classification abilities of the model would be positively impacted even with a small number of parameters.

Given the limitations in terms of computational resources we cannot make use of a large datasets and models with a lot of parameters. In this context, we only rely on data from the Librispeech corpus (a consensual good quality speech records). We propose a comparative study of the input features to assess their ability to discriminate between speech/non-speech on the evaluation data. However, since audio excerpts only contain speech signal, there are chances that our model will learn to detect the signal activity, because it is always concomitant with the speech activity.

2 Experimental protocol

2.1 Data & pre-processing

For this work we use a subset of Librispeech composed of 975 good quality $16kHz$ 16bits signed integer audio sequences. Every sequences are associated with a JSON file containing time annotations of the speech segments which correspond to the individual words (begin/end in second). Audio sequences present variable lengths (see illustration in Figure 1). However, the major part of audio files last up to 15s.

The dataset regroups 34 different speakers that we separate in two different sets. We randomly picked out 23 speakers that will be used for training, while others are kept-out for the evaluation. From the audio waveform we compute a 40 bins mel scaled fbank, concatenated with energy. It will form the 41-dimensional input vector of the model. Input vectors are computed over 25ms length sliding windows with a 10ms shift resulting in variable length input

vectors sequences denoted. From the annotation files we extract the binary labels Y with speech segments $Y = 1$ and non-speech segments $Y = 0$. Since labels are single event boundaries expressed in seconds, we have to take care of the alignment with their corresponding features frames.

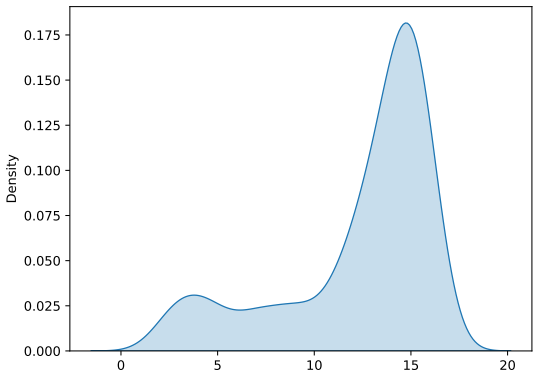


FIGURE 1 – Audio duration distribution

2.2 Model architecture

We use a Convolutional Recurrent Deep Neural Network architecture with parameters constrained to a very small number. Our model consists of a pre-normalization layer followed by two Convolutional blocks, a RNN and two Fully Connected blocks. Each convolutional block operates two 1-D conv, each followed by a normalization layer and a leaky ReLU that are connected to a final dropout. We use two stacked Bidirectional Gated Recurrent Unit for our recurrent network. Finally, the classification part is held by two fully connected blocks composed of linear layers with leaky ReLU, batch normalizations and dropouts. A last linear layer with a single unit is added, since we face a binary classification scheme. The model counts 47.394 trainable parameters in total.

2.3 Training process

We first randomly divide the training data in two sets in order to keep 25% of all training sequences for validation. Input sequences are then zero padded on the right to reach the target size fixed arbitrarily to 1563 frames. It roughly corresponds to 15 seconds of signal. Finally, all data samples are normalized thanks to the mean and standard deviation calculated over the entire training set. We set a binary cross-entropy function as our optimization objective and use the Adadelta algorithm to update the model parameters with initial learning-rate set to 1.0.

The model parameters are all initialized with the Pytorch method, unless for convolutions weights and the RNN hidden states weights for which we use Kaiming and orthogonal initialization respectively. The former is known to improve stability of the activation and reduce chances of gradient vanishing, while the latter prevents gradient exploding/vanishing when recurring along long sequences with RNN. We track training and validation losses through epochs and save the best performing model. In Figure 2 we present a summary of the losses while training over 100 epochs. We use a mini-batch size of 2.

2.4 Evaluation

Depending on the sequences, lots of zeros frames and labels are added to the sequences, due to the padding. We must be careful when analyzing results because it could artificially increase the true negative rate. For the evaluation of the task we refer to usual binary classification evaluation metrics (*e.g.* ROC curve and EER). ROC curve evaluate the system's decision in terms of True Acceptance Rate and False Acceptance Rate. A well performing system would depict a ROC curve that bend toward the top left corner where True Acceptance is maximal and False Acceptance is minimal. In addition we refers to the Equal Error Rate for evaluation as it is an unbiased measure of the performances of the system.

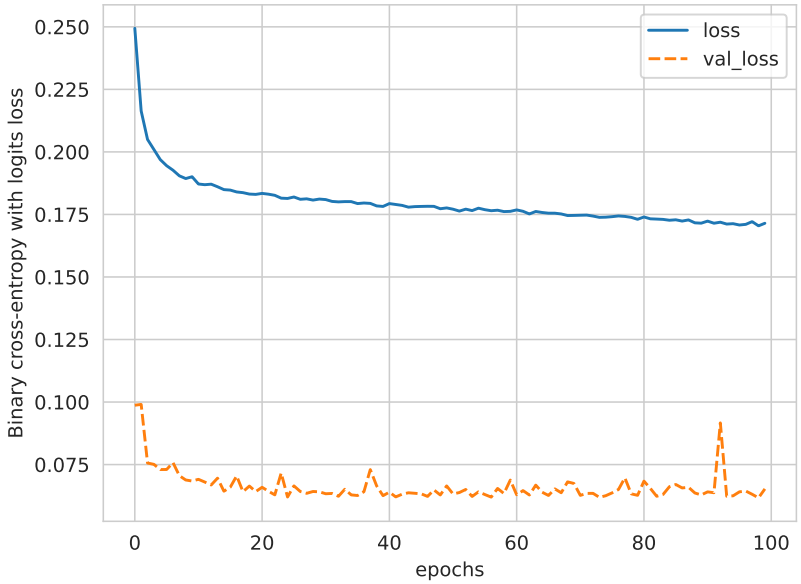


FIGURE 2 – Training loss and validation loss behave surprisingly. The validation is less than the training loss. This is mainly due to the strong regularization employed. Training loss did not improve much, because the small number of parameters constrained the model very much and acts like a regularization factor that prevent overfitting.

3 Post-processing

3.1 Decision

Since the model only predicts the speech probability at frame-level we must add additional operations in order to take a decision. We look at the neighborhood frames probabilities to base our decision. We first apply a smoothing function to the raw probabilities. In this work we considered Exponential Moving Average and Kalman Filter. Once probabilities are smoothed, we compute frames with a probability of speech greater to the deactivation and the activation threshold which are set to 0.55 and 0.3 respectively. Next, we use a sliding window to look at the direct neighborhood of the considered frame

and we set the decision to the minimum value of the 24 consecutive frames, which corresponds to 0.25s of signal. We chose to reduce the window to its minimum value as we observe better results empirically. However we could consider the use of the mean or the maximum aggregation function in order to add a decision fading effect (hangover scheme).

3.2 Segmentation

Segmentation post-processing aims at merging segments that are next to each others and to remove the segments that are too short. We use a merging threshold of $0.022s$, mainly because it corresponds to the minimal tonal sound that a human ear can perceive. In addition, we set the minimum duration of a segment to $0.25s$, otherwise it is deleted.

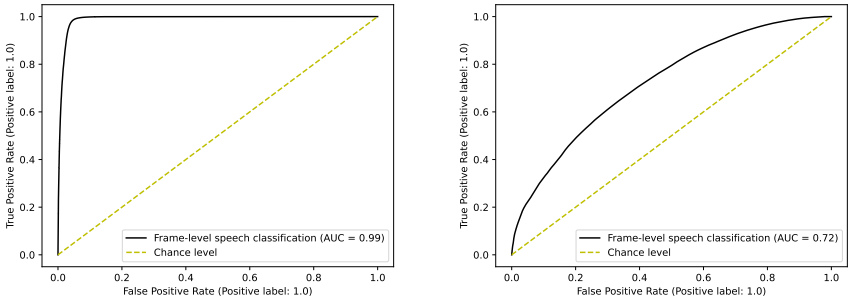
4 Results

In Table 1, we present the results of the input features comparison. According to the %EER, we achieved the best performances with the combination of both Fbank and energy features, while the sole energy gives poor results. In addition, in Figure 3 we illustrate our results by referring to the Receiver Operating Characteristic (ROC) curve obtained during evaluation. We see that the Fbank+Energy system performs almost perfectly.

Features	Loss	%EER (th)	Precision	Recall	F-measure
Fbank	0.103	0.043 (0.81)	0.98	0.96	0.97
Fbank+Energy	0.09	0.037 (0.82)	0.98	0.96	0.97
Energy	0.57	0.525 (0.63)	0.78	0.65	0.71

Table 1 – Features comparison results

To better understand how the system behaves, we illustrate, in Figure 4, a view of the speech probabilities predictions made over a randomly picked-out audio example in comparison to the groundtruth labels. As we can see, the predictions supersede the speech signal activity very well. However, small boundaries between words aren't detected, since time interval is too short.



(a) (b)

FIGURE 3 – ROC curve comparison of Fbank+Energy features (a) and Energy only (b)

After a mere analysis, we found out that the model based on energy-only features gives unexpected speech probabilities. However, due to time limitation we are not able to carry out further analyses. More details are presented in Appendix 5 with the strategy we would adopt.

5 Conclusion & Discussion

In the last decade, deep learning approaches have demonstrated good abilities to perform voice activity detection. Driven by specific application needs, research now tend to focus on 1) the robustness of the model to noise ; 2) low resources computation environment. In this work we propose a convolutional recurrent deep neural network with less than 50.000 parameters to tackle this task. Even if our system showed good performances (%EER < 0.04) on evaluation data, we must be aware that these performances can drastically decline if we would encounter unclean or noisy data. Future works would aim at the improvement of the robustness of the model by employing different approaches : 1) using data augmentation ; 2) using domain adaptation methods and 3) self-supervized approaches.

Data augmentation techniques would consist in adding different kind of noises and reverberations over the speech signal. Moreover, the combination

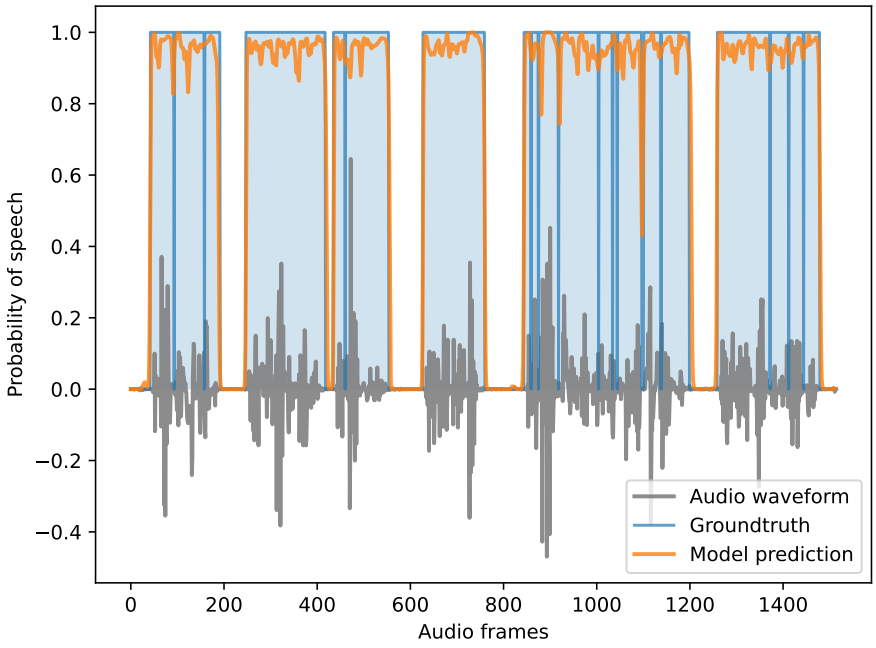


FIGURE 4 – Model predictions made over a random audio example in comparison to the groundtruth labels

of multiple speakers voices in an additional data sample could be done using mix-up techniques. Therefore, we could asses the robustness of the model to detect specific voices in a crowd. Domain adaptation methods have gained a lot of interest in the recent years, especially domain adversarial learning approaches like in [5]. In such methods, model learns discriminative features that are invariant to the domain. It is of great interest for multilingual problematic, but could be experienced for noise robustness as in [4]. Finally, self-supervized approaches would substantially increase the quantity of data available, since it does not necessitate direct annotations. Even if it would require large scale pretraining steps, we would be able to recover a reasonable model size with the help of compression methods like quantisation/pruning, alike knowledge-distillation methods in order to fit into more

a resources constrained environment. However, large scale self-supervized pretraining steps require a lot of care. As it is a very consumptive process, we cannot perform wide grid search to find the most suitable hyperparameters four our model. Thankfully, several methods have been introduced to find a good starting point like warmup. The latter helps the model to slowly adapt (with a low learning-rate) to the data for some steps. Also, it is useful for adaptive optimizer like Adam, because it can now compute its first and second order statistics, without impacting the model weights too much. Moreover, learning-rate is known to scale relatively well regarding to the number of trainable parameters. Thus, initial learning-rate can be determined using a smaller version of models and use scheduling to ensure that model weights continues to vary. Others methods can help the training of large models. Adding residual connections between blocks ensures the propagation of activation through deeper networks. Weights decay like dropout are both regularization techniques that we often use nowadays instead of $l1$ or $l2$ regularization terms. Also, paying attention to the way parameters are initialized can prevent gradient vanishing. Finally, concerning model with a larger set of parameters trained with a large volume of data, we must take into consideration the memory consumption, especially regarding transformer models. Several methods have been developed recently, (*e.g.* sharded data parallelization, mixed precision training, etc.) but it goes out of scope of this work.

Références

[1] T. Hughes and K. Mierle, "Recurrent neural networks for voice activity detection," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7378–7382, IEEE, 2013.

[2] N. Ryant, M. Liberman, and J. Yuan, "Speech activity detection on youtube using deep neural networks.," in *INTERSPEECH*, pp. 728–731, Lyon, France, 2013.

[3] S. Thomas, S. Ganapathy, G. Saon, and H. Soltau, "Analyzing convolutional neural networks for speech activity detection in mismatched acoustic conditions," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2519–2523, IEEE, 2014.

[4] M. Lavechin, M.-P. Gill, R. Bousbib, H. Bredin, and L. P. Garcia-Perera,

"End-to-end domain-adversarial voice activity detection," *arXiv preprint arXiv :1910.10655*, 2019.

[5] Q. Wang, W. Rao, S. Sun, L. Xie, E. S. Chng, and H. Li, "Unsupervised domain adaptation via domain adversarial training for speaker recognition," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4889–4893, IEEE, 2018.

Appendix A

Predictions probabilities remain high but slowly decreased with time as we can see on Figure 5. It is difficult to make any interpretation about this behaviour. Since we monitored good learning metrics during training, we could suppose that model have learnt to detect the global activity (energy) in the remaining signal. First, we could try to reverse the audio signal. Since the energy should not be impacted by this transformation, the observation of similar predictions would point into that direction. Otherwise, we could check whether the size of the sequence have an impact on the prediction or not. Then, we would investigate on what the network have really learnt and especially the convolutional filters. By looking at the kernels of the convolution layers we would be able to see the impulse response of the filter. A FFT would gives the amplitude of the impulse response in the frequency domain. Chances are that the CNN have learnt to emphasize a specific frequency range of the signal. Given that speech activity stand as the only source of energy, the model could focus on the first strongest frequencies it encounters and got stuck by the optimization process in that local optimum.

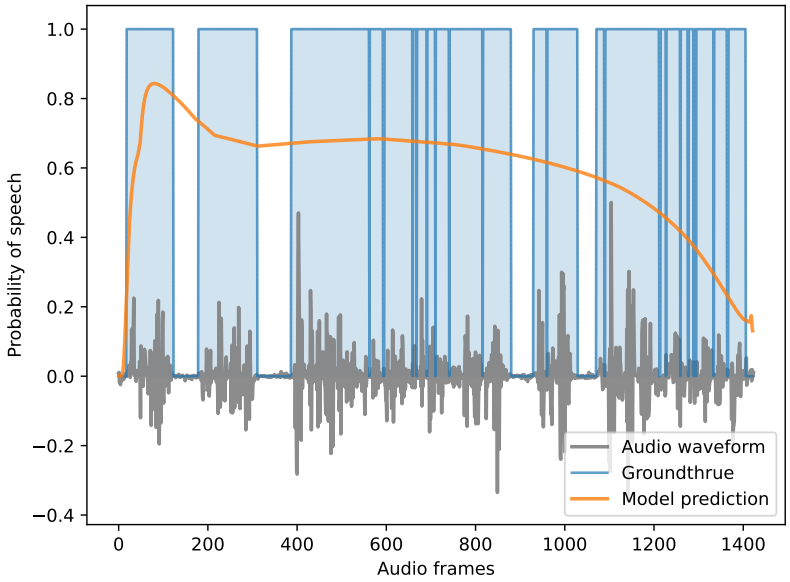


Figure 5 – Energy-only model predictions