Voice activity detection

Tuomas Kaseva, Sirong Huang and Petr Baca

Student ID: 474102, 361150, 645258
Aalto university, Otaniemi, Finland

Voice Activity Detection (VAD) is an important task for speech recognition with wide variety of other applications. In this paper, we first review relevant literature, followed by introducing experiment design and implementation methods. After that, the experiment results are presented and conclusions are drawn based on the feature performance in various noise-conditions. Finally, the limitations and future improvements are discussed.

1 Literature Review

Voice Activity Detection refers to the analysis of an audio signal to determine whether the presence of speech [1].

VAD has a wide variety of applications. The most common examples are discriminative processing of audio signal (speech vs non-speech), activating or deactivating the application (Siri, OK Google), ASR systems for the speaker verification, etc.

Recently, VAD is used also for the blind upmixing or perceptual audio coding. Since the speech is the communication tool, we often need to treat it differently [1]. To provide an particular example, we discuss the case of discontinuous transmission. This method saves data when transmitting and receiving happen simultaneously (typically phone call). Since there is no need to transmit data when one speaker is silent, the transmission can be disabled and the output replaced with artificially generated comfort noise.

Another application of VAD is speech enhancement. When we try to estimate spectral content of the noise to attenuate it, we need to find a spectra of non-speech part of the signal. And as a last example, we might want to use different audio coding for speech and music part of the signal, since speech signal has narrower spectrum content.

Example implementations of VAD include standardized VAD for GSM and VAD in AMR [1]. VAD for GSM refers to an implementation where VAD is used in wireless telephony. In the implementation VAD is used to detect speech pauses using information of spectral distortions between consecutive frames. If the spectral distortion is deep below desirable threshold enough, VAD concludes that a non-speech segment is present.

VAD in AMR, or utilizing VAD in Adaptive Multirate speech codec, has generally two choices of VAD implementations [2]. However, both of the options are based on a same principle. Speech and non-speech segments are located with comparing the input level of the signal and background noise level [1]. In background noise estimation, a tone detection function is used in order to find signals with periodical structure [1].

VAD implementations are essentially based on information encoded in various features of audio signals. These features are computed from overlapping segments of the sound signal, usually 10-50ms long [1]. Often, better results are achieved when more features are computed and utilized. Each feature represents certain properties of the signal, and they can be divided into the following categories.

The simplest type of features are based on the intensity of the signal. When there is no or only silent background noise, VAD becomes a signal activity detection task. Most simple one is root-mean-square computation. However, these simple features does not work well when treating noisy audio signals. Another equally simple feature is zero-crossing rate.

It is prominent time-domain feature, which is easy to compute. It uses the fact that harmonic signal crosses zero less often than noise which is often a set of random numbers. This feature is the least CPU demanding, though, also least robust and could be easily mislead.

More advanced features describe the tonality of the signal. Since human speech produces a range of tonal content as fundamental frequency, formants or upper harmonics, tonality features contain plenty information which can be exploited in order to distinguish between speech and non-speech signal segments. For example, autocorrelation, average magnitude difference and linear prediction error can be used for this purpose. However, those features are not efficient when extracting speech from another periodic signal, e.g. music.

There are so-called spectral shape features that can represent complex signals, using MFCC (mel-frequency cepstral coefficients) [3][4] or other measurements. Taking advantage of all the information contained in this high dimensional feature as spectral spread, we can also distinguish music from the speech.

The earliest form of VAD algorithm is manually tuned decision threshold based on certain features such as energy threshold, zero crossing rate. This type of algorithmn is very computational inexpensive but requires laborious manual tuning of large numbers of parameters beforehand to achieve optimal error rate.

Since VAD in its nature is as a binary classification problem, more general purpose machine learning classification algorithms can be applied to VAD implementation. Examples of classification algorithms that have seen success in VAD research are K-Nearest Neighbors, Support Vector Machines, Bayes classifier, Random Forest classifier, etc. [1] However, since each frame of audio signal is treated separately and independently, these methods do not utilize the important temporal continuity information contained in audio signals, leading to inferior performance compared to sequential algorithms.

Recent research has shown that sequential algorithms such as Recurrent Neural Networks can outperform traditional VAD methods tremendously, due to their capability to model audio signal as sequence and exploiting the temporal structure of audio signal.[5]

When VAD implementation is evaluated, the starting-point is to compare the VAD results with an original signal. Generally, one has a frame based VAD classification with each frame classified as speech or non-speech. After classification, classified frames are summed and summation is divided with overall number of frames. As an outcome, a quantified percentage of wrongly labeled frames is obtained. This evaluation method is called a mean error rate (MER) [1].

MER itself does not contain all the relevant information about speech activity. In addition, the evaluation of VAD should also take in consideration the segments where the noise is detected as speech or speech detected as noise [6]. Especially the later is very undesirable since it can produce clipping. Detecting noise as speech is also problematic in some cases such as speech coding where it is desirable to keep a bitrate as low as possible.

2 Implementation

This section presents our VDA implementation, which can be described as a machine learning algorithm for binary classification. As already discussed in literature review, general VAD implementations are based on using features of speech in order to determine if fragments of signal are speech or not. The principal is same also in our implementation. First, signal is windowed and partitioned into frames. Secondly, different numbers and combinations of features are computed for each frame. Finally, the labels (speech or non-speech) are determined for each frame using support vector machine based on the features of the frame. In order for the classifier to work, it is trained beforehand with labeled training data.

In this section, we first present the features and discuss their relevance, then describe our choice of classifier - support vector machine, and finally discuss the structure of VAD implementation in more detail.

2.1 Features

Following section explains, how used features work. It begins with concluding properties of speech itself, and then continues with the features themselves. For the implementation, zero-crossing rate, energy value, one-lag autocorrelation and MFCC were chosen as features.

2.1.1 Speech properties

First, basic properties of speech will be discussed. This is essential for further understanding the way, how we treat the signal with various features. In typical speech recognition task, we have a signal containing the speech and the noise. Moreover, there can be some else background noise, e.g. music or other speech, but those cases are rather complicated and this summary won't cover them.

Noise is just a set of random number values. If it is normalized, there are samples with values within [-1,1] in time domain. The most common noise found in the environment has spectral distribution of the energy of the pink noise, which means, that the energy decays with a ratio of -3db/octave. [7].

In the contrary to this, speech, as well as any other sound perceived by humans somehow meaningful (e.g. music) has the structure of a harmonic signal. This comes with various properties. The harmonic signal can be described by its frequency, resp. period. It has also its amplitude. It is just a summation of many sine waves with different frequencies and phases. Once the signal has its frequency, resp. fundamental frequency, its pitch can be perceived.

2.1.2 Zero-crossing rate

Zero crossing rate works with the periodicity of the speech signal. Any harmonic, resp. periodic signal has its period and frequency. The lower the frequency, the higher the period of the signal. Since the speech signal has the majority of frequency content between 80 Hz and 7 kHz, it is rather low-frequency signal. Zero-crossing rate computes the moments, when the signal crosses zero within fixed frame. The lower the frequency, the lower the zero-crossing rate. So, with setting a proper threshold, we can assume, that if the ZCR is low enough, the frame contains harmonic signal - speech. [1]

$$zcr = \sum_{n=1}^{N-1} [sign(s[n]s[n-1]) < 0] \tag{1}$$

2.1.3 Energy value

When assuming the condition with high signal-to-noise, resp. speech-to-background ratio, measuring the energy works really well. Most typically, when the speaker is close to the microphone, his voice is the strongest input signal captured and then, we can simply measure the energy of the signal. If it reaches the threshold, we assume, that speech is present. However, noise, or any other loud unwanted signal ruins the functionality of the feature. In good condition, it is the simplest and best working one.

$$e = \frac{1}{N}\sqrt{\sum_{i=1}^{N-1} s[i]^2} \tag{2}$$

2.1.4 One-lag autocorrelation

The same property as in ZCR is used is One-lag autocorrelation feature. This is, though, more robust to noise and background not-harmonic unwanted signals. The autocorrelation of harmonic signal is close to 1 and it is close to -1 in case of noise. The sum of the noise with itself delayed picture is attenuated or stays on the same level. This property is also used when recording the sound in room with the stereo techniques. The direct wave is summed, because it is correlated in both microphones. Reflections are attenuated, or stays at the same level. [7]

However, this feature uses the one-lag autocorrelation. In the case of harmonic signals, it is highly probable to get a positive result from following equation:

$$r[1] = \sum_{n=1}^{N-1} s[n]s[n-1] \tag{3}$$

since the ZCR is low, the following two samples are most likely of the same sign. In the case of random numbers (noise) this is not true and we often multiply an negative sample to the following one, which is positive (or vice-versa). The result is determined by the autocorrelation coefficient. When the contribution of positive values is greater than those, which is negative, the coefficient is also positive.

2.1.5 Mel-Frequency Cepstral Coefficients

The most sophisticated feature used in this project, are Mel-Frequency Cepstral Coefficients. Nowadays, this is very popular feature to examine the inspected signal, since it provides very useful and clear data. The motivation to evolve MFCCs is rare - it combines both simulation of human perception and adaption the coefficients to the machine learning needs. Computing the MFCCs has few steps. First (this step is optional) the pre-emphasis filter is applied. Previously, this step was essential to avoid numerical issues in FFT computations, but nowadays most of the implementations of FFT are resistant to this issues. Next, the signal is framed and windowed. Usually, the frames are 20-40ms long and about 50% overlapping. Each frame is windowed, usually using hamming window. Then, FFT of each frame is computed. Those results are filtered by Mel-frequency filter banks. Those filters are triangular filters with varying width, to mimic the perceiving of human hearing. They roughly correspond to critical bands. However, those coefficients are highly correlated, which is not usable in several machine learning applications. To decorrelate them, DCT is applied to obtain the MFCCs. The final step is choosing (for example) 12 most important coefficients, since the coefficients obtained from high frequency bandwidths are less important in ASR tasks, and taking mean of them. [8]

2.2 Support Vector Machine

For the purpose of this study, we want to compare VAD performance with different features in various noise conditions. Therefore we need an algorithm which works well with all the features presented and is easy to train. Support vector machine is a robust and fast classifier that have been reported to be effective in a wide range of applications including VAD

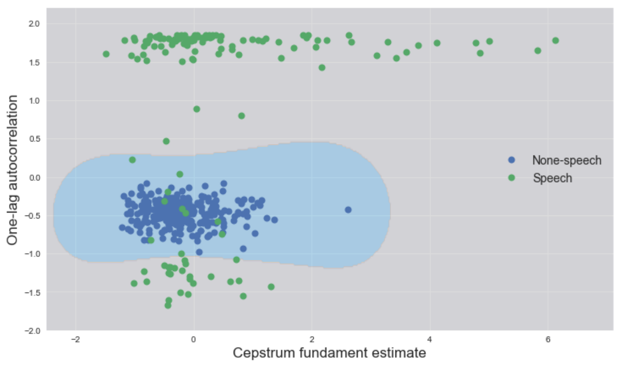

Figure 1: *SVM with two features on train and test data with medium noise*

[9, 10, 11]. Moreover, it provides an equal ground for the comparison of all the features used in this study including MFCC, and can be trained without difficult manual tuning of numerous parameters, making it a solid algorithm for comparative study of VAD features in various noise conditions.

Support Vector Machine is a binary classifier that finds the decision boundary that separates two classes best by maximizing the margin between two classes. Figure 1 is an example of SVM decision boundray with two features.

After tuning parameters using 5-fold cross validation, the best SVM hyperparameters setting for this experiment is Non-linear Gaussian SVM with C=1 regularization.

2.3 Structure Of Implementation

In the beginning of this section, a brief introduction of a structure of the implementation was given. This subsection explains the implementation more thoroughly including creation of data, training and testing. Essentially, the whole VAD system is generated with three following steps.

1. **Preprocessing.** Firstly, feature matrices and label vectors are computed from training and test datasets. Each dataset composes of J number of wav-files which are divided into frames using Hann-window function. For each frame, features are calculated and labels determined. The procedure has to be repeated 15 times for each wav-file, since using 4 features we have 15 different combinations (when 1, 2, 3 or 4 features can be exploited). From now on, we refer to the combinations as **feature sets**. In addition, a noisy test set is also created with adding Gausian noise with magnitude -50 dB to the original test set.

2. **Training.** Secondly, we designed two types of noise conditions: white noise, non-stationary noise. With white noise, we exposed four different levels of white noise to either training or testing data: noise-free, low, medium, high white noise, resulting in 16 different experiments. With non-stationary noise, we experimented on 3 types of real life noise conditions: crowd speaking in the background, car passing by, keyboard typing. With 15 different feature sets that can be used, the performance of resulting numerous experiments will be examined closely in the test session.

3. **Testing.** Finally, VAD systems trained with various feature sets and noise condition data will be evaluated on clean or noisy data. The evaluation is performed with comparing generated test labels with real test labels. Based on the observations, a best performing model configuration will be chosen and the VAD system is completed. This part of implementation will be enlighten in more detail in next section.

3 Evaluation

Intuitively, the performance of VAD implementation is based on how many times the classifier predicts correct label for each frame. However, as discussed in literature review, this comparison does not give all the information of the quality of VAD system. In addition, one has to deduce, **which are the circumstances when error is present**. For this reason, three different error measurements are used in the evaluation:

1. **MER** (Mean error rate)
2. **NDASER** (Noise detected as speech error rate)
3. **SDANER** (Speech detected as noise error rate)

Each of these error rates are based on comparing the actual labels and predicted labels. MER is calculated with formula

$$\text{MER }\% = \frac{1}{N}\sum_{i=1}^{J}\frac{M_i}{N_i}, \tag{4}$$

where N denotes number of frames in all test set wav-files, M_i denotes number of incorrectly labeled frame and N_i number all frames of ith wav-file. Similarly, NDASER can be generated from equation

$$\text{NDASER }\% = \frac{1}{N}\sum_{i=1}^{J}\frac{K_i}{N_i} \tag{5}$$

and SDANER from

$$\text{SDANER }\% = \frac{1}{N}\sum_{i=1}^{J}\frac{L_i}{N_i}, \tag{6}$$

where K_i and L_I denote incorrectly labeled frames in ith wav-file when original label is noise and when then label is speech, respectively. Naturally, $M_i = K_i + L_i$

For each of the feature sets, using original and noisy test sets, three error measurements values are calculated. The results are presented in the following subsection.

3.1 Experiment Results

This section presents the results of different experiments and the performance measurements of different features in various conditions.

Figure 2, 3 shows two experiments with white noise of different volume added. When trained and tested with noise-free data, all feature errors are acceptable. We can see that the *energy* level feature performs with higher speech as noise error. This is probably caused by the parts of speech such as plosives or fricatives which are basically silence, but human consider it as speech - it is part of it. We could reduce this error by applying smoothing.

As we add louder white noise to test set, the error rate rapidly increases. Noise-as-speech error is higher than speech-as-noise error, which is caused by noise being included also in training data. We can see, that *MFCC* remained quite robust and well-performing even in this condition.

Figure 4 shows the result of noise-free SVM tested on none-stationary noise, where *MFCC* keeps its superior performance, but *one-lag autocorrelation* and *zero-crossing* have much inferior performance. We can see that the *energy* error is still quite well-performing, since the speaker is way closer to the capturing microphone (although this is specific to this experiment's data), the energy of the desired speech grows when he starts speaking.

Including the crowd noise also to testing data (Figure 5) rapidly improves the behaviour of all the features and *MFCC* starts working really well. *Energy* level feature also set its threshold more properly, so it performs with almost no noise as speech error.

4 Conclusions

VAD is a challenging and important task in various applications. In this project, four features' performance in voice activity detection were examined in various noise conditions. The examined features were zero-crossing rate, energy value, one-lag autocorrelation and MFCCs. Those were tested with several noise conditions including: noise-free, white noise of low, medium, high volume, and non-stationary noise such as crowd speaking in the background.

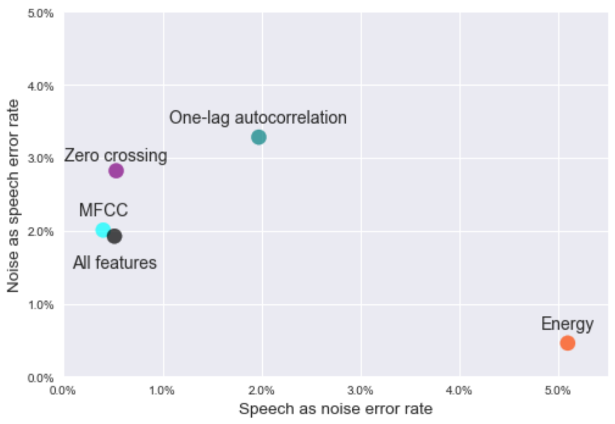

Figure 2: *SVM trained and tested on noise-free data*

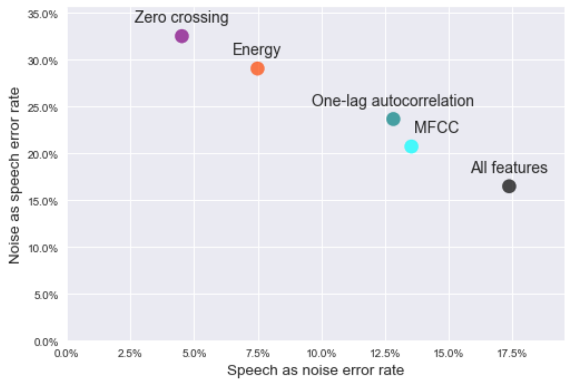

Figure 3: *SVM trained and tested on loud white noise data*

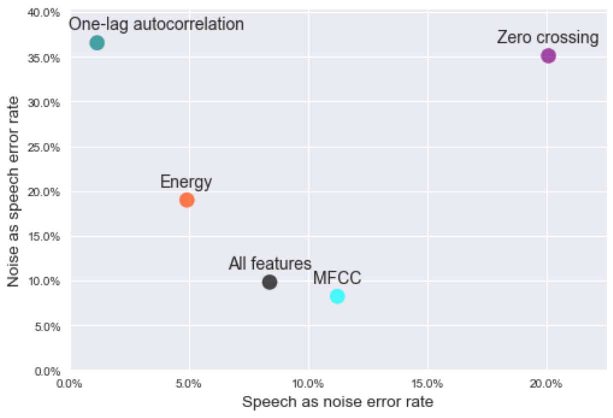

Figure 4: *SVM trained on noise-free data, tested on crowd speaking noise data*

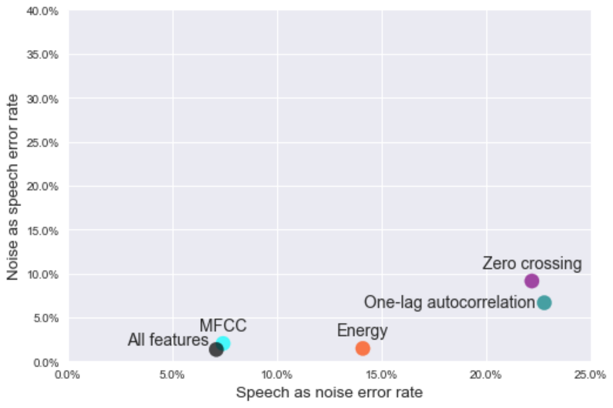

Figure 5: *SVM trained and tested on crowd speaking noise data*

Generally when less noise is present in test data, VAD classifiers performs much better. However, to our surprise, when training the classifier with nosier training data, the classifier performs worse most of the time. Although, when adding the same noise present in the test data to the training data, the accuracy of corresponding classifier increased tremendously. Therefore, we conclude that VAD is a highly application dependent task that requires data collected specifically for certain application to achieve optimal performance.

As for feature performance, all the features we've chosen were valuable in certain situations, but we can say that, in most tasks MFCC performs the best. However, strictly speaking, there is no single "best" feature that outperforms the others in every situations. It is often a trade-off between noise-as-speech error or speech-as-noise error, depending on which information is prioritized to preserve or eliminate.

To conclude the experiments, VAD application is a highly application dependent task both in terms of feature choice and training data choice. To achieve optimal performance, VAD should use a feature set that minimize the type of error that the application desires, and train the classifier with noises present in the actual application.

5 Limitations

There are a few limitations about this experimental study. First of all, the data consists of a single male speaker recording at a fixed distance towards the microphone. Therefore, the data does not represent different speakers' identities or varying speaker distances.

Secondly, the VAD performance of the optimally configured classifier can be further improved by implementing smoothing techniques to reduced speech clipping and random noise during silent intervals. For the simplicity of feature comparison, Support Vector Machine is used as classification algorithm of choice. More advanced algorithms such as recurrent neural network can further improve performance by exploiting the temporal structure information of audio signals.

References

[1] Tom Bäckström. Speech Coding: With Code-Excited Linear Prediction. 2017, Springer Publishing Company, Incorporated.

[2] Etienne Cornu, Hamid Sheikhzadeh, Robert L Brennan, Hamid Reza Abutalebi, Edmund CY Tam, Peter Iles, and Kar Wai Wong. Etsi amr-2 vad: evaluation and ultra low-resource implementation. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 2, pages II–841. IEEE, 2003.

[3] P. Davis, S. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. pages 357–366. IEEE, 1980.

[4] A. Acero X. Huang and H. Hon. Spoken language processing: A guide to theory, algorithm, and system development. Prentice Hall, 2001.

[5] Thad Hughes and Keir Mierle. Recurrent neural networks for voice activity detection. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 7378–7382. IEEE, 2013.

[6] DK Freeman, G Cosier, CB Southcott, and I Boyd. The voice activity detector for the pan-european digital cellular mobile telephone service. In *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, pages 369–372. IEEE, 1989.

[7] Ville Pulkki. Communication Acoustics: An Introduction to Speech, Audio and Psychoacoustics. 2014.

[8] Haytham Fayek. Speech processing for machine learning: Filter banks, mel-frequency cepstral coefficients (mfccs) and what's in-between. 2016.

[9] Tomi Kinnunen, Evgenia Chernenko, Marko Tuononen, Pasi Fränti, and Haizhou Li. Voice activity detection using mfcc features and support vector machine, 2007.

[10] Javier Ramírez, P Yelamos, Juan Gorriz, and José Segura. Svm-based speech endpoint detection using contextual speech features. 42:426 – 428, 04 2006.

[11] Dong Enqing, Liu Guizhong, Zhou Yatong, and Zhang Xiaodi. Applying support vector machines to voice activity detection. In *Signal Processing, 2002 6th International Conference on*, volume 2, pages 1124–1127. IEEE, 2002.