Abstract

This paper demonstrates how feedback from a speech recognizer can be leveraged to improve Voice Activity Detection (VAD) for online speech recognition. First, reliably transcribed segments of audio are fed back by the recognizer as supervision for VAD model adaptation. This allows the much stronger LVCSR acoustic models to be harnessed without adding computation. Second, when to make a VAD decision is dictated by the recognizer not the VAD module, allowing an implicit dynamic look-ahead for VAD. This improves robustness but can be gracefully reduced to meet latency requirements if necessary without requiring retraining/retuning of the VAD module. Experiments on telephone conversations yielded a 6.7% abs. reduction in frame classification error rate when feedback was applied to HMM-based VAD and a 4.2% abs. reduction over the best baseline system. Furthermore, a 3.0% abs. WER reduction was achieved over the best baseline in speech recognition experiments.

Index Terms: voice activity detection (VAD), speech segmentation, speech recognition

1. Introduction

Voice Activity Detection (VAD) is a classical problem to detect the presence of speech in a noisy signal and has become an essential part of systems for speech coding, speech enhancement, speech recognition and echo cancellation. This paper focuses on VAD for online large vocabulary continuous speech recognition (LVCSR). Unfortunately, VAD for LVCSR is often an afterthought, and off-the-shelf VAD components designed for other tasks are simply picked up and marginally tweaked. The fact that there is a very powerful speech recognizer in the loop is often ignored. This work seeks to leverage that speech recognizer and demonstrates how feedback from the recognizer can be used to improve the robustness of VAD.

Many VAD algorithms target applications for embedded processors such as mobile phones. As such, they use simple algorithms with limited computation and memory requirements. Typically features like linear prediction coefficients, energy, zero crossing rate, cepstral features, and periodicity are extracted and then a statistical or rule-based classifier is used to make VAD decisions. The standardized ITU-T G.729 Annex B [1] VAD algorithm uses feature combination to further improve robustness. Another standard organization, ETSI, adopts adaptive multi-rate (AMR) VAD [2] and advanced front-end VAD [3] for feature extraction and distributed speech recognition. A Gaussian statistical model and likelihood ratio test together with decision-directed parameter estimation for adaptation was used in [4]. Model refinements such as the soft decision rule [5], discriminative weight training [6] and alternate distributions (Laplacian and Gamma [7]) have also been shown to yield improvements. Unfortunately, none of these leverage the speech recognizer to improve VAD robustness.

The obvious way to leverage the speech recognizer is to use its acoustic models. For example, a common approach used in multi-pass LVCSR is to run a recognition pass with a narrow decoding beam and then use a smoothed version of the transcript as the segmentation. LVCSR acoustic models use millions of parameters for representing speech, silence, and in some systems even noise, and are typically trained up on a large amount of data to capture speaker and environment variability. Using these models directly for VAD however is computationally and memory intensive, and thus inappropriate for applications such as distributed speech recognition.

Instead, this paper proposes using feedback from the recognizer to guide VAD. First, reliably transcribed segments of audio are fed back by the recognizer as supervision for VAD model adaptation. This allows the much stronger LVCSR acoustic models to be harnessed without adding computation. Second, when to make a VAD decision is dictated by the recognizer not the VAD module, allowing an implicit dynamic look-ahead for VAD. This improves robustness but can be gracefully reduced to meet latency requirements if necessary without requiring retraining/retuning of the VAD module.

Feedback is already commonly used in commercial cascaded multi-layer VAD components. For example, a more accurate, slower VAD layer using a different set of features may feedback its supervision to a faster, less accurate VAD layer. Feedback is internal to the VAD component and the models used tend to be simplistic to constrain computation. The contribution in this work is to leverage external feedback from a speech recognizer and allow those more powerful models to guide VAD adaptation without introducing latency or significant computation.

This paper is organized as follows. Section 2 explains two approaches to leveraging recognizer feedback in VAD. Section 3 then gives a concrete example for applying this feedback to Hidden Markov Model (HMM) based VAD. Section 4 reports on experiments for VAD accuracy and speech recognition word error rate (WER). Finally conclusions are drawn in section 5.

2. Recognizer Feedback For VAD

Leveraging recognizer feedback instead of the recognizer itself means that the VAD module remains simple and compact. This is important when running on low-resource devices such as mobile phones (backed of course by a recognizer running on a remote server). An obvious limitation of leveraging LVCSR feedback is that an LVCSR system is needed and thus this approach does not make sense for non-LVCSR applications such as speech coding. Two types of feedback are discussed here: 1) directing VAD model adaptation with feedback and 2) directing look-ahead with feedback.

2.1. Directing Model Adaptation

Most VAD algorithms perform online adaptation of their models. Typically this is incremental and unsupervised where previous VAD decisions are used to guide a subsequent adaptation. Here it is proposed to use the recognizer transcript as supervision for...
adaptation. Audio is first recognized using LVCSR until end-of-utterance is detected by VAD. The recognition transcript is then confidence-scored (e.g. by using word posterior scores from the recognition lattice) and frames in confident words are fed back to the VAD module as speech frames. Similarly frames falling in confidently transcribed silence/noise words are fed back as silence/noise frames. The VAD models are then adapted and recognition and VAD continue with the new VAD models. Figure 1 demonstrates this idea.

The motivation here is to leverage the much more powerful speech recognition acoustic models for reliably identifying speech, silence and optionally noise. The confidence scoring restricts this to only the reliably transcribed parts of the audio. These confident parts tend to be further away from speech/silence/noise boundaries which means the models are adapted to non-transitory data. Unfortunately, it is a reality that many LVCSR systems do not have an explicit noise model. Furthermore many recognizers are trained on only clean speech and thus have no representation of noise. Ideally, the LVCSR acoustic models should be retrained to include mixed noise condition data and an explicit noise model. However, this is unrealistic, as it may negatively impact the overall task WER and does not make sense solely to improve VAD. Instead, the approach used here is to simply ignore the problem. That is, if a noise model is available, use it and feed back frames labeled as noise. Otherwise, simply feed back silence-labeled frames and treat those as noise. This rather harsh assumption was found to be workable as demonstrated by the experiments in section 4.

2.2. Directing Look-Ahead

Making a VAD decision using a look-ahead window typically improves accuracy at the expense of latency. The amount of look-ahead is normally fixed and VAD models are trained/tuned using that look-ahead. Changing the look-ahead (e.g. to adjust latency for an application) usually requires retraining VAD models or adjusting decision rules, which is inconvenient. The approach proposed here uses a delayed VAD decision mechanism to allow a dynamic look-ahead that does not require model retraining.

Conceptually an LVCSR system can be partitioned into 2 threads: a front-end thread for feature extraction and signal processing, and a back-end decoder that processes features and generates a recognition lattice. The front-end simply processes incoming audio samples and generates feature vectors. Typically, VAD decisions are also emitted with a vector - this is referred to as a synchronous VAD decision.

Instead, it is proposed to use an asynchronous approach. At each frame $t$, multiple hypotheses regarding the classification of the frame are maintained - no classification decision is made. VAD decisions are instead made by rescoring those multiple hypotheses at the time the VAD decision is actually required, which may be some time after $t$. This is shown in figure 1. The requirement of multiple hypotheses restricts the applicability of this type of feedback to VAD algorithms that use multiple hypothesis or probabilistic decoding.

For example, consider a LVCSR system combined with a hypothetical Finite State Transducer (FST) driven VAD. For each frame the state of the FST is simply updated and no VAD label is assigned. Furthermore, assume the LVCSR requires a look-ahead of $\Delta$ frames for delta feature computation and short-term feature normalization. Hence, in order to decode frame $t$, the front-end, and therefore the VAD FST, is advanced to frame $t + \Delta$. When the decoder retrieves the feature vector for frame $t$ it also requests the VAD decision. At that point, the VAD FST trace-back (which has been stepped so far to frame $t + \Delta$) is examined and the best decision for frame $t$ is returned. This implicitly introduces look-ahead smoothing into the VAD decision since frames beyond $t$ could alter the best trace-back path. If the recognizer look-ahead is somehow reduced, there will simply be less benefit from look-ahead smoothing and no changes will be required to models or decision rules.

Since the VAD label for a frame is dynamic, sampling the label again further in the future will likely result in a more correct label. This opens opportunities for cascading the risk of VAD decisions. For example, an application may declare end-of-utterance given only a few frames of silence (to give quick response), but may wait for many more frames before declaring end-of-conversation. Badly labeled frames will be automatically corrected as more frames are decoded and thus an incorrect end-of-conversation decision may be avoided even if a bad end-of-utterance decision was made.

3. Augmenting HMM-based VAD With Recognizer Feedback

A concrete implementation of HMM-based VAD with recognizer feedback is given here to further demonstrate leveraging recognizer feedback. It should be noted though that other VAD algorithms can also be augmented with the above feedback, for example model adaptation in G.729B [1] or Sohn’s VAD [4].

3.1. HMM structure for VAD

In HMM-based VAD, the Maximum Likelihood criteria is used to align a multi-state fully-connected HMM against input speech. The resulting state alignment then gives the VAD class for each frame. In the simplest case, a 2-state HMM is used, one each for speech and non-speech. Formally then the VAD decision for $y_t$, the $t$th frame of the observed noisy signal is given by the label of the occupied state at that time, and governed by the hypotheses:

$$H_0 \text{ (speech absent )} : y_t = n_t$$
$$H_1 \text{ (speech present )} : y_t = x_t + n_t$$

where $n_t$, and $x_t$ are noise and active speech respectively. The HMM states are modeled by statistical distributions. Here the multi-variate Gaussian is used, with a diagonal covariance matrix under the weak assumption of independent Gaussian random variables

$$p(y_t|H_k) = \frac{1}{\sqrt{(2\pi)^m|\Sigma_k|}} \exp \left(-\frac{1}{2} (y_t - \mu_k)^T \Sigma_k^{-1} (y_t - \mu_k) \right)$$

where $\mu_k$ and $\Sigma_k$ are the parameters of the speech/non-speech states. The initial values of these parameters are trained on reference segmented noisy data.
3.2. VAD Decoding

Online VAD is performed using the Viterbi algorithm to align the HMM states to the input signal. The LVCSR front-end feature extractor and back-end decoder operate concurrently. The following step sequence is executed per frame:

1. The LVCSR requests a speech feature vector \( o_t \) for frame \( t \).
2. To compute \( o_t \), the front-end extracts up to \( o_{t+\Delta} \), where \( \Delta \) is governed by the requirements for delta/acceleration computation, short-term feature normalization or application latency.
3. Correspondingly VAD is decided to frame \( t + \Delta \).
4. The LVCSR decoder requests the VAD decision for frame \( t \).
5. The best-path in the Viterbi network at time \( t + \Delta \) is extracted, and the path is traced back to find which state was active at time \( t \). Optionally smoothing is applied on the VAD best path decisions to remove short speech/non-speech segments. The label of that state is emitted as the VAD decision at time \( t \).

An end-of-utterance is declared when \( N \) consecutive frames of non-speech are found. Requesting the VAD label for frame \( t \) at a later time may result in a different label if the Viterbi best path has changed. It was found useful to add a minimum duration constraint to states: 0.05s for non-speech and 0.1s for speech.

3.3. Model Adaptation

At the end of an utterance, the LVCSR emits the speech recognition lattice. A confidence score is then computed per word on the lattice best path using the well-known word posterior

\[
p(w_i|Y) = \sum_{\pi} p(w_i|\pi) \sum_{\pi} p(\pi|Y) P_{LM}(\pi) \tag{4}
\]

where \( w_i \) is word edge on the best path, \( \pi \) is a path in the lattice, \( P_{LM}(\pi) \) is the language model probability of the word sequence path \( \pi \), \( p(\pi|Y) \) is the acoustic likelihood of the observation sequence \( Y \) for the utterance, and \( \alpha \) is the acoustic scaling factor. Simply put, this is the sum of all paths that span edge \( w_i \) divided by the sum of all lattice paths. The confidences of edges with the same word label and time boundaries are combined. Then, all frames in the best path edges with confidence above a threshold are selected as adaptation frames. Specifically, for the best path edges with confidence above a threshold \( \tau \),

\[
\sum_{t} p(\pi|Y) P_{LM}(\pi) \sum_{t} p(\pi|Y) P_{LM}(\pi) \tag{4}
\]

where \( \pi \) is the new distribution estimated from all frames seen since \( \tau = 0 \) (not just those for the current utterance), \( N(\tilde{\mu}_k, \tilde{\Sigma}_k) \) is the new distribution, and \( \tau \) is the MAP adaptation weight. The adapted VAD models are used from then on. Models may be reset to the original trained models at the end of a conversation or speaker interaction.

4. Experiments

Experiments were conducted on telephone speech to evaluate VAD frame classification error rate and speech recognition WER.

4.1. Experimental setup

Noisy recordings were created by mixing clean signals with 4 types of real-world noise samples from the AURORA-2 database [8]: car (stationary), restaurant (babble), street (non-stationary) and subway (periodic). Noise samples were re-sampled to 8kHz and then partitioned into training and testing noise samples using a 50% split. Then noise-corrupted training and test sets were artificially created by additive mixing of a weighted noise signal with a clean speech signal. Signal-to-Noise Ratio (SNR) was defined as the ratio of the active speech energy of clean signal \( s(t) \) to the energy of the corresponding segments from the noise \( n(t) \).

\[
SNR = 10 \log_{10} \frac{\sum_{t=1}^{T} s^2(t) r(t)}{\sum_{t=1}^{T} n^2(t)p(t)} \tag{8}
\]

Here \( r(t) \) was the active speech mask from the reference segmentation: \( r(t) = 1 \) when the clean signal was active and 0 otherwise.

Training speech was sourced from 352 conversation sides from the Switchboard-1 telephone conversation corpus [9]. These were randomly split into 16 subsets with 22 sides each and then mixed with the 4 training noises at 4 different SNRs respectively: 0dB, 10dB, 20dB and clean. The noise mixing weight was calculated per side and the active speech mask \( r(t) \) was taken from the speech utterance boundaries given in the reference word transcriptions of the data. This reference segmentation did include non-speech frames at the start and end of speech segments, but it was considered acceptable to allow these since silence at speech recognition boundaries does not hurt WER. The speech and noise HMM states were then trained using the AURORA multi-condition training procedure in [8].

Evaluations were done on 40 conversation sides from the HUB5 Switchboard 2000 Evaluation set. Noise-corrupted speech was created using the same procedure as used for training speech, to give 1 clean and 12 noisy versions of each conversation side. Furthermore, sides from the same conversation were combined to a single mixed channel and then noisy versions were created as well. This gave in total 60 files (40 single-sided conversation sides and 20 mixed conversations) with 13 noise conditions giving a total of 780 test files.

All recognition experiments used a 72-mixture triphone AM ML-trained on 2000h of Switchboard and Fisher data, a 3,157-gram language model trained on a mixture of telephone conversations, broadcast news lectures, and a 50k vocabulary. This system achieved a 19.3% WER on the HUB5 Switchboard 2000 Eval. set using multi-pass recognition with speaker adaptation. However, in the experiments below the recognizer was run in real-time single-pass online mode. The recognizer used a look-ahead window of 0.5s for feature normalization and the same for look-ahead in feedback-augmented HMM-based VAD (fbHMM) experiments. A confidence threshold of \( \rho = 0.8 \) was used to select frames feedback adaptation.

The VAD baselines used for comparison were G.729B (the speech coding standard ITU-T G.729B [1]), AFE (ETSI AFE [3] used in the distributed speech recognition system in AURORA), and Sohn (a well-known statistical model based method [4]).

4.2. Voice Activity Detection Results

VAD was evaluated using the average classification error rate shown as \( E = \frac{1}{N} \sum_{k} E_k / \bar{N}_k \), where \( \bar{N}_k \) and \( E_k \) denote the total number of frames and the number of incorrectly classified frames respectively for a test file \( k \). Reference frame classification was taken from the speech segmentation of the evaluation data word transcriptions. Since speech recognition does not require precise speech/non-speech boundaries, real time hangover smoothing was
Table 1: Frame classification error rate (%) on Switchboard 2000 Eval telephone conversations for various noise conditions and VAD methods.

<table>
<thead>
<tr>
<th>Noise</th>
<th>SNR</th>
<th>G.729B</th>
<th>AFE</th>
<th>Sohn</th>
<th>fbHMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car</td>
<td>0dB</td>
<td>52.3</td>
<td>30.2</td>
<td>30.1</td>
<td>20.9</td>
</tr>
<tr>
<td></td>
<td>10dB</td>
<td>29.7</td>
<td>16.6</td>
<td>5.2</td>
<td>6.1</td>
</tr>
<tr>
<td></td>
<td>20dB</td>
<td>13.3</td>
<td>15.1</td>
<td>3.3</td>
<td>3.5</td>
</tr>
<tr>
<td>Restaurant</td>
<td>0dB</td>
<td>30.6</td>
<td>30.0</td>
<td>32.8</td>
<td>25.0</td>
</tr>
<tr>
<td></td>
<td>10dB</td>
<td>22.5</td>
<td>24.0</td>
<td>11.0</td>
<td>15.6</td>
</tr>
<tr>
<td></td>
<td>20dB</td>
<td>19.8</td>
<td>22.8</td>
<td>7.4</td>
<td>9.1</td>
</tr>
<tr>
<td>Street</td>
<td>0dB</td>
<td>29.0</td>
<td>25.1</td>
<td>23.9</td>
<td>14.8</td>
</tr>
<tr>
<td></td>
<td>10dB</td>
<td>16.2</td>
<td>21.1</td>
<td>13.5</td>
<td>7.0</td>
</tr>
<tr>
<td></td>
<td>20dB</td>
<td>13.3</td>
<td>21.0</td>
<td>11.2</td>
<td>3.9</td>
</tr>
<tr>
<td>Subway</td>
<td>0dB</td>
<td>29.7</td>
<td>26.7</td>
<td>36.3</td>
<td>19.9</td>
</tr>
<tr>
<td></td>
<td>10dB</td>
<td>25.1</td>
<td>24.9</td>
<td>7.3</td>
<td>6.2</td>
</tr>
<tr>
<td></td>
<td>20dB</td>
<td>22.2</td>
<td>24.6</td>
<td>3.0</td>
<td>3.5</td>
</tr>
<tr>
<td>Clean</td>
<td></td>
<td>12.1</td>
<td>24.2</td>
<td>9.4</td>
<td>3.5</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>24.3</td>
<td>23.5</td>
<td>14.9</td>
<td>10.7</td>
</tr>
</tbody>
</table>

Table 2: Influence of recognizer feedback signals on frame classification error rate. DLook was directed look-ahead, Rec.Adapt was recognizer directed adaptation, None was without any feedback.

<table>
<thead>
<tr>
<th>fbHMM</th>
<th>None</th>
<th>DLook + Rec.Adapt</th>
<th>DLook</th>
<th>Rec.Adapt</th>
</tr>
</thead>
<tbody>
<tr>
<td>G.729B</td>
<td>16.7</td>
<td>10.7</td>
<td>13.5</td>
<td>12.4</td>
</tr>
<tr>
<td></td>
<td>24.3</td>
<td>23.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

applied. Thus, speech segments less than 0.05s were changed to non-speech and non-speech segments less than 0.1s were changed to speech, and then a collar of 0.5s was added to any remaining speech segments.

Table 1 shows comparative results for various VAD schemes and in different noise conditions. The fbHMM achieved an average classification error rate of 10.7%, which was 4.2% absolute better than the best baseline (Sohn). In fairness though, the fbHMM system was trained on in-domain data while the baselines were existing reference systems. Sohn’s VAD generally seemed to do better in the 10-20dB SNR range and error rates rapidly increased outside this suggesting that Sohn’s system had been tuned to this SNR range.

4.3. Effect of Feedback

Table 2 shows the results for experiments to quantify the contribution of recognizer feedback to HMM and G.729B VAD (similar experiments could not be done on the Sohn and AFE due to lack of source code access). Without feedback, the error rate of HMM VAD was 16.7%. Directing VAD look-ahead using feedback decreased this to 13.3%. In contrast, directing model adaptation using recognizer feedback gave an error rate of 12.4%. Combining both gave a final error rate of 10.7% demonstrating that the feedback mechanisms were complimentary. In constrast, feedback only reduced error rate for G.729B VAD from 24.3% to 23.7%. This was less than expected and was because the speech/silence supervision from the recognizer was not precise. The G.729B models were not as forgiving as the HMM models of poor adaptation supervision.

4.4. Effect on Speech Recognition

The VAD methods were also evaluated for online LVCSR. WERs on the evaluation set are shown in Table 3. For the ideal reference (Ref.), G.729B, AFE and Sohn’s systems, VAD was run on an entire conversation side and then a single pass of speech recognition was run on each segment. A fixed recognizer and associated models was used in all cases, only the VAD result was changed each time. In contrast, VAD and recognition were done in a single pass for fbHMM. Only clean and 20dB noisy speech WERs are reported as the recognition models were not appropriate for low SNR conditions. The fbHMM system achieved the best WER of 28.3% in clean condition and 35.7% in 20dB noisy condition, and outperformed the best baseline system by 3.1% and 2.8% absolute respectively. However, there was still a significant gap of 2.9% and 3.3% compared to using the reference segmentation.

5. Conclusions

The paper has demonstrated that VAD error rates can be significantly reduced by leveraging speech recognizer feedback. The proposed approach allows the considerably stronger speech recognition models to be harnessed for VAD model adaptation without increasing runtime. Furthermore, it allows the recognizer to dictate when to make a VAD decision, enabling an implicit dynamic look-ahead that improves robustness but can be gracefully reduced to meet latency requirements. Although recognizer feedback can be applied to a variety of VAD schemes, the experiments reported here focused on HMM-based VAD. It was shown that HMM-based VAD frame classification error rates were reduced by 6.7% absolute to 10.7% for a telephone conversation evaluation set, and outperformed the best evaluated baseline approach by 4.2% absolute. Furthermore, speech recognition experiments achieved a reduction of 3.0% abs. in WER over the best baseline when leveraging feedback for VAD. Overall, it has been demonstrated that exploiting the speech recognition context for VAD is indeed beneficial.

6. References

[3] ETSI Speech processing, transmission and quality aspects (STQ); Distributed speech recognition; Advanced front-end feature extraction algorithm; Compression algorithms, ETSI ES 202 050 v1.1.5, 2007.