VOICE ACTIVITY DETECTION IN NOISY ENVIRONMENTS

Nicklas Hansen (s153077), Simon Holst Albrechtsen (s134868)

Technical University of Denmark

Guidance provided by Retune DSP

ABSTRACT

Automatic speech recognition (ASR) systems often require an always-on low-complexity Voice Activity Detection (VAD) module to identify voice before forwarding it for further processing in order to reduce power consumption. In most real-life scenarios recorded audio is noisy and deep neural networks have proven more robust to noise than the traditionally used statistical methods. This study investigates the performance of three distinct low-complexity architectures – namely Long Short-Term Memory (LSTM) Recurrent Neural Networks (RNN), Gated Recurrent Unit (GRU) RNNs and an implementation of DenseNet. Furthermore, the impact of Focal Loss (FL) over the Cross-Entropy (CE) criterion during training is explored and findings are compared to recent VAD research.

Using a 72-hour dataset built from open sources with varied noise levels, 12 Mel-frequency Cepstral Coefficients (MFCC) as well as their derivatives in a temporal context of 900 ms, a GRU-RNN with 30.000 parameters achieves an Area Under Curve (AUC) of .991 and a False Acceptance Rate (FAR) of 3.61% given a False Rejection Rate (FRR) fixed at 1%. Focal Loss is found to improve performance slightly when using focusing parameter $\gamma = 2$ and performance improvements are observed for all three architectures when their number of parameters is increased, which suggests that network size and performance can be viewed as a trade-off.

It is observed that in a high-noise environment, Convolutional Neural Networks (CNN) struggle compared to pure RNNs where a 10.000 parameter LSTM-RNN achieves a FAR of 48.13% for fixed FRR at 1% compared to 58.14% for a DenseNet of comparable size.

Index Terms— Voice Activity Detection, CNN, RNN, Noisy Environments, Low Complexity

1. INTRODUCTION

Increasingly many embedded systems – such as smart home devices and speakers – utilize ASR as an integral part of the user interface. As embedded systems typically have limited processing power and are battery-driven, there is a demand for low-complexity ASR systems that are robust to noise.

VAD is the task of determining whether a voice is currently present or not and is typically applied as a first step in real-time, always-on ASR systems in order to decrease overall power consumption by avoiding further processing if no voice is present.

To provide a good user experience it is crucial for a VAD module to have a very low FRR, which is challenging in noisy environments (i.e. most real-life scenarios). If speech is momentarily rejected by the VAD module, the following step in the processing pipeline may fail to recognize e.g. a wake-word employed to wake up the full ASR system. On the other hand, if a VAD module achieves an infinitesimal FRR but has a high FAR, the power consumption of the VAD process may very well exceed the amount of power it was supposed to save.

Historically, VADs have been built using statistical methods but they tend to perform poorly in noisy environments. Google WebRTC VAD [1] is a publicly available and widely used VAD built as a Gaussian Mixture-Model. Modern VADs are typically based on deep neural networks as they have proven far superior to statistical methods in less ideal environments. One research group has applied a LSTM-RNN to VAD in popular Hollywood movies with great success [2].

Recent efforts shift their attention to deep dilated CNNs [3]. Using a 36-layer dilated CNN, they achieve a FAR of 5.67% compared to 6.66% for a comparable LSTM-RNN with a fixed FRR of 1% for both architectures. Although such deep architectures are shown to produce excellent results for VAD they are impractical for use in embedded real-time ASR systems as they are immensely resource-intensive due to their large size.

In this paper we briefly present the two mentioned deep learning approaches and introduce the reader to the production of a large dataset built from open sources, which is discussed in section 3 and section 4. FL – an alteration to the CE criterion – is introduced in the following section and the paper then proceeds to present three distinct deep learning architectures proposed in this study. Experimental details are provided in section 7 and results from said experiments are discussed in section 8. Finally, opportunities for future work are briefly touched upon in section 9.

Implementation is available at https://github.com/s153077/vad.

2. STATE OF THE ART

Temporal modeling of voice using recurrent networks and convolutional networks has previously been explored in the literature with great success. In this section, two approaches to modern VADs based on deep learning are briefly presented.

2.1. Eyben's approach

The VAD proposed by Florian Eyben, et. al. [2] is based on a recurrent network architecture using LSTM cells in two distinct setups: 1) a single recurrent layer and 2) 3 recurrent layers. Rather than applying the model to raw audio, 18 MFCCs are extracted using a frame size of 25 ms and their derivatives are computed to produce a total of 36 acoustic features. Continuous but varied noise is applied to a dataset consisting of 26 hours of informal speech. Computed frames are inputted to the network in sequences of length 50 (equal to a 1.25 s window). The exact number of parameters has not been disclosed in neither of the two networks.
They find that their approach outperforms statistical methods by far in terms of accuracy – especially on samples that contain noise. The small and large networks achieve an AUC of .951 and .961, respectively, on their test set compared to a mere .821 by the best-performing statistical method proposed.

2.2. Shuo-Yiin's approach

Shuo-Yiin, et. al. [3] explore temporal modeling using a CNN for VAD to achieve state-of-the-art results.
Contrary to [2], they compute 40 MFCCs (also using a frame size of 25 ms) but refrain to use their derivatives. As such, [3] uses a total of 40 acoustic features. Like the previous study, varied background noise is applied to clean speech to produce a simulated noisy dataset. In this study, a stunning 18.000 hours of audio is used as training data and evaluation is carried out on 15 hours of similar data. Computed frames are inputted to the network in sequences of length 270 (equal to 6.75 s).
By utilizing dilation, gating and residual connections they build a massive 36-layer deep CNN (and an additional fully-connected layer and an output layer) containing approximately 400.000 parameters. They find that their model achieves a FAR of 5.67% compared to 9.76% for a conventional CNN for a fixed FRR at 1%. Additionally, they build a stacked LSTM of similar size, which achieves a FAR of 6.66% under the same circumstances.
As such, they reinforce the idea that both convolution and recurrent networks can be used to capture temporal patterns in acoustic data. A significant drawback from their proposed architectures; however, is that they are immensely resource-intensive and cannot feasibly be used to capture voice in a real-time scenario without introducing a significant amount of latency.

3. DATASETS

LibriSpeech ASR corpus [4] is an open-source dataset consisting of 360 hours of clean speech recordings in English. This study uses a subset (36 hours, 10%), sampled uniformly at random from the training set of the dataset.
In order to mimic real-life scenarios with background noise the QUT-NOISE database [5] is included to provide "natural" noise to the clean speech, which provides various types of noise that are common to experience in a user's daily life [5]. From the QUT-NOISE dataset, two sources of noise have been excluded – the cafe and living room sets – as they contain subtle background speech. Such speech can cause incorrect labelling of audio as it technically is voice.

4. PRE-PROCESSING

The LibriSpeech corpus is merged into a single audio stream of clean speech data. WebRTC VAD [1] [6] is used to label the audio using a 30 ms frame size and a sensitivity of zero. The dataset is sliced into snippets of varying length between 1000 ms and 5000 ms in a uniform distribution. An equal amount of slices containing silence is created using a similar varying length. These two slice sets are merged and shuffled together to create a single dataset with a random pattern of speech and silence. This amounts to a total of 72 hours of audio with a 50/50 distribution of voice and silence.
Three different levels of noise are added to the audio in order to simulate a varied environment. All noise is normalized and applied on top of speech audio at the following noise levels: none, low (-15 dB) and high (-3 dB).

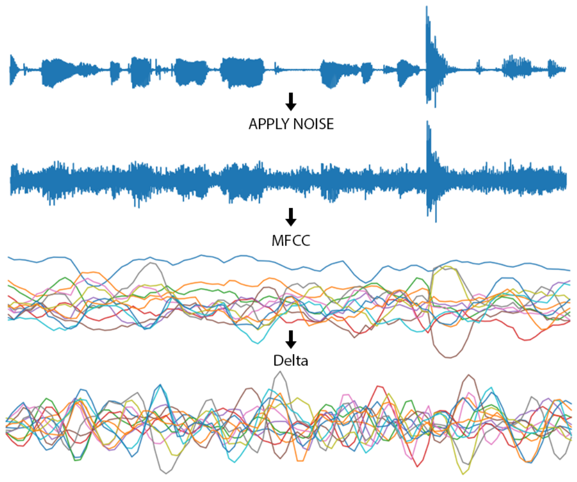

Fig. 1: Key steps in pre-processing of samples and feature extraction.

In some instances of VAD applications the raw audio input is applied a short time fast Fourier transform (STFT) and a 2D convolutional layer to extract the features for use in the subsequent layers.
The problem with such an approach in the context of this study is that it produces a three-dimensional feature-set, which is computationally expensive and therefore not suitable for low-complexity applications of VADs.
To overcome this problem MFCCs with a frame size of 30 ms along with the delta of these coefficients are computed to create 24 features per audio frame resulting in a two-dimensional feature-set.
As it is possible to compute MFCCs efficiently using several different approaches, it is deemed an appropriate approach for this study given the low-complexity constraints [7] [8] [9].

5. FOCAL LOSS

To maximize performance in noisy environments, it is desired to weight poorly-classified samples more than well-classified samples while still exploiting the availability of easy samples. This is achieved by application of *Focal Loss* [10] as the criterion during training, which adds a modulating factor $(1-p_t)^\gamma$ to the CE loss function:

$$\text{FL}(p_t) = (1-p_t)^\gamma \ \text{CE}(p_t) = -(1-p_t)^\gamma \ \log p_t \qquad (1)$$

where $p_t = p$ for $y = 1$ and $1 - p$ otherwise (for a prediction p). γ is the *focusing parameter* and it serves to down-weight the influence of well-classified samples (where $p_t \gg 0.5$). For a prediction $p_t = 0.95$, the loss for FL using $\gamma = 2$ is 400 times smaller than CE, forcing the optimizer to focus far more on poorly-classified samples to minimize loss. For $\gamma = 0$ the modulating factor equals 1 and the loss is thus equal to that of CE.

6. PROPOSED ARCHITECTURES

Three distinct architectures for VAD have been adopted in this study: a LSTM-RNN, a GRU-RNN with features derived from convolutional layers, and a compact implementation of DenseNet [11].
To evaluate model performance in a constrained parameter environment, all three architectures have been evaluated at two fixed parameter counts: 10.000 and 30.000 parameters (approximately). As such, a total of 6 distinct low-complexity models are proposed and the influence of parameter space is investigated. The two network sizes are referred to as *small* and *large* in the following.

6.1. Long Short-Term Memory cells

The LSTM-RNN models serve as a performance *baseline* and consists of a single unidirectional LSTM-layer (30 cells) as well as one or two fully-connected (FC) layers, depending on network size. The LSTM cell architecture adopted in this study is as originally proposed by [12] and as implemented by [2]. Inputs are formatted such that a cell input x_k is a 24-dimensional feature vector consisting of the MFCCs and deltas corresponding to the kth frame (time-step).

6.2. Gated Recurrent Units and Gated Convolution

The second architecture is a GRU-RNN consisting of three or four gated convolutional layers (depending on network size), a single unidirectional GRU-layer (30 cells) and finally one or two FC layers followed by a softmax output layer.
A GRU implements its gating mechanisms similarly to LSTM but lacks a memory unit, which exposes the full hidden state to the cell [13]. Empirically, GRUs have been shown to perform on par with LSTM on a number of sequence modeling problems within the audio domain [14] and the lack of memory units decrease the number of parameters as well as computational complexity. No decrease in performance was found on the problem addressed in this study, thus GRUs were favored over LSTM cells.
One-dimensional convolution with zero-padding and fixed kernel size of 3 (90 ms) is applied along the temporal axis to capture short-term patterns in time. Thus, each feature represents a channel in the convolutional layer.
Gated convolution is similar to the gating mechanisms found in LSTMs and GRUs in that an input is convoluted independently with two different sets of filters – regular filters and gate filters. The hyperbolic tangent function is applied to output from regular filters and output from gates pass through a sigmoid function. The hadamard product (element-wise multiplication, denoted $\odot$) of the two resulting matrices is then computed and forwarded as the final output of that layer. The full computation of hidden state h_k given input x_{k-1} and weights $W_{f,k-1}, W_{g,k-1}$ can thus be written as Equation 2:

$$h_k = \tanh(W_{f,k-1} * x_{k-1}) \odot \sigma(W_{g,k-1} * x_{k-1}) \qquad (2)$$

Refer to Figure 3 for an illustration of the gating mechanism. Due to the limited number of convolutional layers as well as the fact that shallow CNNs are less prone to vanishing gradients [3], residual connections were not found to improve model performance and have thus been abandoned.
Batch normalization and dropout ($p = .2$) is applied whenever appropriate.

6.3. DenseNet

The third architecture proposed is an implementation of DenseNet [11] that has been adapted to the temporal nature of VAD. DenseNet attempts to maximize the information flow through the CNN layers by connecting the CNN layers via concatenation and therefore improve the feed-forward properties of the CNN layers. The initial layer consist of

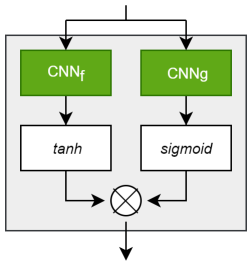

Fig. 2: Gated convolution as used in the GRU-RNN.

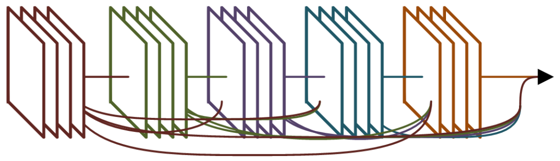

Fig. 3: Illustration of a dense block as used in the DenseNet.

a dilated CNN layer with max-pooling, which allows for a broad capture of the moving window while keeping the parameter count low. The output from the dilated CNN layer is forwarded into two dense blocks connected by a transition layer, which tries to diminish over-fitting. The output of the last dense block is passed through a single CNN layer with max-pooling and finally a single softmax layer.
The main difference between the small- and large DenseNet is the amount of layers in each dense block and the number of channels used in the CNN layers.

7. EXPERIMENTAL DETAILS

Models are trained on all three noise levels using a batch size of 2048, 30 frames (i.e. a window of 900 ms) and 24 features derived from MFCC and their derivatives. Models are trained for 15 epochs using FL as criterion and a γ-value of 0 (regular CE) or 2; whichever yields best accuracy on the validation set is kept. If the accuracy is the same, $\gamma = 0$ is used. The value of γ is selected based on results in [10] as well as a preliminary evaluation. One limitation of our study is that samples of different noise levels are not shuffled prior to training, which may diminish the effect of FL to some extent.
Adam [15] with weight decay 1e-5 and initial learning rate of 1e-3 is applied as optimizer for the LSTM-RNN and GRU-RNN architectures while Stochastic Gradient Descent (SGD) with a learning rate of 1 and momentum 0.7 is applied to DenseNet as it yields better results.
Models are evaluated on a test set generated from the same source but with no overlap in speech. For each of the three noise levels, ROC-Curves are computed and the Area Under Curve (AUC) is used as primary performance metric. Additionally, FAR is calculated for fixed FRR at 1% for comparison to [3].

Noise: None	**LSTM-RNN**	**GRU-RNN**	**DenseNet**
Small	5.11%	4.71%	6.86%
Large	4.34%	**3.61%**	5.75%

Noise: Low	**LSTM-RNN**	**GRU-RNN**	**DenseNet**
Small	17.18%	15.58%	24.45%
Large	16.84%	**11.14%**	23.15%

Noise: High	**LSTM-RNN**	**GRU-RNN**	**DenseNet**
Small	**48.13%**	63.10%	58.14%
Large	51.75%	58.99%	61.02%

Table 3: FAR for fixed FRR at 1% for each of the three noise levels. Bold font indicates the lowest FAR at each level.

8. RESULTS

In this section, we present our experimental performance results of the three architectures proposed for VAD in noisy environments. In subsection 8.1, the impact of FL is investigated. In the following subsections, the focusing parameter γ that yields the best results on a validation set is selected.
Table 1 shows the selected value of γ for each of the proposed architectures and network sizes.

γ-value	**LSTM-RNN**	**GRU-RNN**	**DenseNet**
Small	0	2	2
Large	2	2	2

Table 1: Selected value of focusing parameter γ for each of the proposed architectures and network sizes.

8.1. Results using Focal Loss

Experimentally, it is found that whether FL has a significant impact on performance or not appears to depend on the particular architecture. Table 2 highlights the positive impact of FL on the large LSTM-RNN while no improvement is found on the small network. While the table only shows AUC for the high-noise test set, minor improvements are evident in the other noise levels as well for the large model while the small model seemingly remains unaffected by FL.
Further research is needed to determine the exact impact of FL when applied to VAD, but it shows potential.

LSTM-RNN	**CE**	**FL**
Small	.965	.965
Large	.962	.969

Table 2: AUC on highest noise level test set for two sizes of a LSTM-RNN trained with and without FL ($\gamma = 2$).

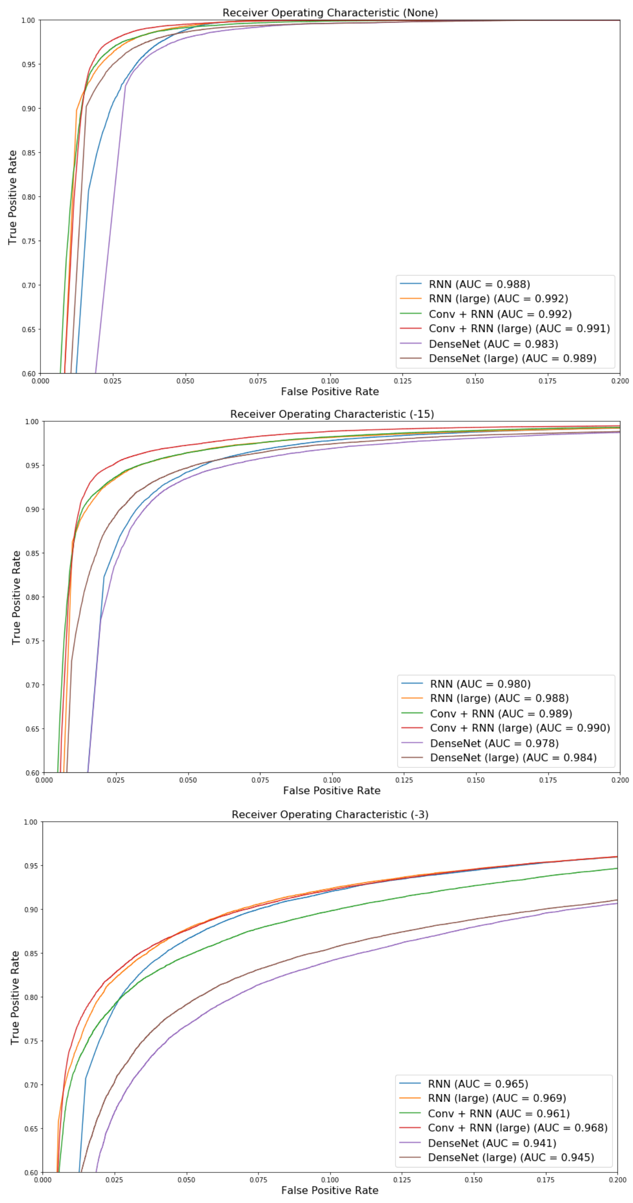

Fig. 4: ROC-Curves for each of the three noise levels: none, low (-15 dB) and high (-3 dB).

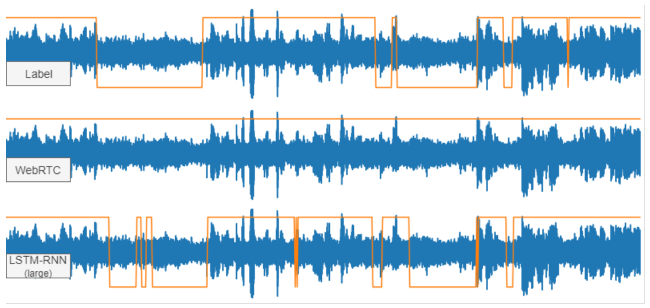

Fig. 5: Top to bottom: Label and predictions by WebRTC and large LSTM-RNN, respectively, on high noise level (20 s). Labels are generated by WebRTC in a clean environment.

8.2. Results from parameter constraints

Figure 4 depicts the ROC-Curves for each of the three noise levels. When using AUC as measure for performance, the two recurrent architectures do not benefit much from increased network size on the test sets without noise and with little noise. DenseNet, on the other hand, consistently improves its AUC across all noise levels as the number of parameters increases. Therefore, it is speculative whether its performance will continue to improve when further increasing network size. On the highest noise level, LSTM-RNN and GRU-RNN also see significant improvements to AUC as their network size is increases, for which we can ask the same question.
It should be noted that – since labeling is automated and thus imperfect – there is a natural limit to how well the networks can perform on a given test set. As most models achieve an AUC around .992 with no background noise, this may be near the upper limit for performance given the imperfect labeling.

8.3. Results using proposed architectures

It can be concluded that all 6 proposed models achieve a high AUC on the test sets. The small DenseNet and small LSTM-RNN performs slightly worse on the test set without background noise and with a small amount of noise, respectively (refer to Figure 4). As noise level increases, the true positive rate (TPR) decreases rapidly while the false positive rate (FPR) remains relatively stable. Therefore, it can be deduced that a VAD module based on the proposed architectures tend to more often accept noise as voice in high-noise environments than environments with little noise, which is consistent with theory. It is observed that the large LSTM-RNN and the large GRU-RNN achieve the highest AUC on the high-noise test set, which suggests that large recurrent networks may be more robust to noise than a DenseNet of comparable size.
As already deduced, it can be observed that FAR increases drastically for all proposed models as noise level increases for a FRR fixed at 1%. Interestingly, the large GRU-RNN achieves the lowest FAR without noise and with low noise – 3.61% and 11.14%, respectively – but is greatly outperformed on the highest noise level by both small and large LSTM-RNNs (48.13% and 51.75%, respectively). FAR does not appear to decrease as DenseNet increases in size, and recurrent networks might thus be favored in high-noise environments. In fact, as all 4 models using CNNs perform poorly on the high-noise dataset in terms of FAR it hints that convolution may be more prone to classifying noise as voice compared to recurrent networks for this particular problem.

8.4. Relation to other studies

While it is not possible to use the two studies, [2] and [3], as direct benchmarks due to different datasets and hyperparameters, findings can be compared in a general sense.

[2] concluded that recurrent networks are more stable in noisy environments against statistical methods, which has been confirmed qualitatively in this study. As shown by [2], LSTM-RNN is a good architecture for VAD and has been reproduced with great success.
While architectures differ, [3] proves that CNNs are viable for VAD. The DenseNet implementation of this study confirms that convolution is viable by itself and that (assuming low levels of noise) performance scales with network size. Likewise, utilizing CNNs and RNNs in conjunction has been found to yield the best results in a parameter-constrained setting, achieving a FAR of 3.61% for fixed FRR at 1% with no noise and just 30.000 parameters. Comparatively, [3] achieves a FAR of 5.67% for fixed FRR at 1% with an unknown level of noise, 400.000 parameters, more features (40 vs. 24), far larger temporal context (6.75 s vs. 0.9 s) and a dataset 250 times the size of the one used in this study. As such, all of the three architectures proposed can be considered viable low-complexity models in environments with moderate noise levels.

9. OPPORTUNITIES FOR FUTURE WORK

Results from this study provide opportunities for further research. FL has demonstrated potential but an even more varied dataset is needed (both in terms of noise sources and noise levels) for better judgment and the dataset should be shuffled to take full advantage of FL.
It is evident that slight performance improvements occur on samples with low levels of noise when convolutional layers are applied before the time series is fed into a recurrent network. It could be interesting to investigate why convolutional layers do not appear to improve performance significantly on high-noise samples. Furthermore, results from experiments in this study demand research on the application of recurrent convolutional neural networks (R-CNN) as proposed by [16] for VAD.

10. CONCLUSION

Based on the conducted experiments, it can be concluded that CNNs, RNNs and a mixture of the two are viable architectures for low-complexity VAD in noisy environments. MFCCs have proven to be compact and reliable acoustic features and the proposed architectures are capable of accurately classifying voice despite a relatively small temporal context (0.9 s). The GMM-based WebRTC is a good public API for VAD on clean speech but is of little to no use on noisy audio. Therefore, it may be suitable for automated labeling but not as a low-complexity alternative to deep neural networks in noisy environments.
It has been shown experimentally that Focal Loss can have a minor positive impact on performance (AUC of .962 vs. .969 for CE and FL using $\gamma = 2$, respectively, for training of a large LSTM-RNN and testing on a high-noise set) but a larger impact would likely be observed on a dataset with shuffled noise levels and more varied samples in terms of speech and sources of noise.
Based on observations made for different network sizes of each of the three proposed architectures it can be concluded that increasing the number of parameters have a positive impact on performance. Increasing network size from 10.000 to 30.000 for a GRU-RNN decreases FAR for fixed FRR at 1% from 4.71% to 3.61% for clean speech, which is a remarkable improvement. As such, network size and performance is an important trade-off when developing VAD modules.
While a combination of CNNs and RNNs yields the best performance results on sets with no noise or low levels of noise, a small LSTM-RNN scores significantly better on high-noise tests with a FAR for fixed FRR at 1% of 48.13% compared to 61.13% and 58.14% for GRU-RNN and DenseNet, respectively, of comparable size. Therefore, it can be speculated – given a low-complexity constraint – whether R-CNNs are more robust alternatives to regular CNNs in environments with high levels of noise, while maintaining its edge over regular RNNs on lower noise levels.

11. REFERENCES

[1] Google, “Webrtc,” https://webrtc.org, visited 22/10/2018.

[2] Florian Eyben, Felix Weninger, Stefano Squartini, and Bjrn Schuller, “Real-life voice activity detection with lstm recurrent neural networks and an application to hollywood movies,” 2013.

[3] Anshuman Tripathi, Aron van den Oord, Bo Li, Gabor Simko, Oriol Vinyals, Shuo yiin Chang, and Tara Sainath, “Temporal modeling using dilated convolution and gating for voice-activity-detection,” in *ICASSP 2018*, 2018.

[4] Daniel Povey Vassil Panayotov, Guoguo Chen and Sanjeev Khudanpur, “Librispeech asr corpus,” 2015.

[5] R. Vogt D. Dean, S. Sridharan and M. Mason, “Qutnoise databases and protocols,” 2010.

[6] “Python interface to the webrtc voice activity detector,” 2018.

[7] Mohammed Bahoura and Hassan Ezzaidi, “Hardware implementation of MFCC feature extraction for respiratory sounds analysis,” in *2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA)*. 6 2013, IEEE.

[8] Swapnil D. Daphal and Sonal K. Jagtap, “DSP based improved speech recognition system,” in *2012 Inter-*

national Conference on Communication, Information & Computing Technology (ICCICT). 10 2012, IEEE.

[9] Haofeng Kou, Weijia Shang, Ian Lane, and Jike Chong, "Optimized MFCC feature extraction on GPU," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. 6 2013, IEEE.

[10] Tsung-Yi Lin, "Focal loss for dense object detection," 2 2018.

[11] et. al. Gao Huang, "Densely connected convolutional networks," 12 2016.

[12] Sepp Hochreiter and Jrgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[13] et. al. Kyunghyun Cho, "Learning phrase representations using rnn encoderdecoderfor statistical machine translation," 9 2014.

[14] et. al. Junyoung Chun, "Empirical evaluation ofgated recurrent neural networkson sequence modeling," 12 2014.

[15] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," 12 2014.

[16] Ming Liang and Xiaolin Hu, "Recurrent convolutional neural network for object recognition," 2015.