ONLINE WORD-SPOTTING IN CONTINUOUS SPEECH WITH RECURRENT NEURAL NETWORKS

Pallavi Baljekar [†] [‡], *Jill Fain Lehman* [‡], *Rita Singh* [†] [‡]

[†] Carnegie Mellon University, Pittsburgh, PA, USA
[‡] Disney Research, Pittsburgh

ABSTRACT

In this paper we introduce a simplified architecture for gated recurrent neural networks that can be used in single-pass applications, where word-spotting needs to be done in real-time and phoneme-level information is not available for training. The network operates as a self-contained block in a strictly forward-pass configuration to directly generate keyword labels. We call these simple networks causal networks, where the current output is only weighted by the the past inputs and outputs. Since the basic network has a simpler architecture as compared to traditional memory networks used in keyword spotting, it also requires less data to train. Experiments on a standard speech database highlight the behavior and efficacy of such networks. Comparisons with a standard HMM-based keyword spotter show that these networks, while simple, are still more accurate.

Index Terms— Continuous speech, Online word-spotting, Speech recognition, Recurrent neural networks, Gated networks

1. INTRODUCTION

One of the key problems in speech recognition is that of word spotting – detecting whether and where in a recording a specified word occurs. The task itself may take two forms. In one, the word to be spotted is specified only as text. Here word spotting must employ a speech recognition based formalism: the spotter must convert the text to an appropriate sequence of phonemes, and the recording must be searched for the sequence of phonemes in the word [1]. The second form of word spotting is example based: a set of examples of the word are provided. From these the spotter must learn to detect the word in the audio. Unlike the text-based approaches, where the basic patterns recognized are phonetic and words must be built over them, the example-based methods generally model the word in its entirety. While a text-based method can be very flexible (keywords can be changed easily), example-based retrieval can be much more accurate, *e.g.* [2][3].

In this paper we address the problem of example-based word spotting. Although the literature on example-based spotting is significantly smaller than the literature on text-based specification, there have nevertheless been several methods proposed. Many of these attempt to mimic the text-based detection mechanism – attempting to derive a phoneme sequence or lattice, which is then used along with the phonetic models in an automatic speech recognizer [4]. Other methods explicitly build detectors or classifiers for the words themselves, based on a variety of instantaneous and long-term features, without explicitly considering the phonetic composition [5]. The latter approach is particularly useful when one does not have access to a large vocabulary system; even otherwise, it can in fact be more accurate than phoneme-lattice-conversion based methods.

Lately, neural-network based techniques have been found to be highly effective for *speech recognition* tasks, with a number of different approaches employing different network architectures [6]-[8]. Of these, the gated recurrent neural networks (RNN) or their variants which have some kind of memory or recurrence structure built-in have been most successful. One reason for their success can be attributed to their ability to capture the clear temporal structure present in speech signals. Conventional speech recognition systems attempt to capture this using a hidden Markov model. HMMs effectively quantize the underlying state space for the speech and represent the dynamics by characterizing the stochastic transitions through these states. RNN models characterize the dynamics more directly, through first- or higher-order recurrence [9]. However modeling temporal structure through direct recurrence between hidden units in a network tends to be ineffective – learning of the recurrence suffers from the well-known "vanishing-gradient" (or exploding-gradient) problem [10]. The gated RNN or more specifically the long-short-term memory (LSTM) neuronal structure solves this problem by carrying temporal recurrence through gates and storing it over a longer time period in its memory structure called the *Constant Error Carousel (CEC)* [11]. By combining the outputs of a gated RNN which works on vectors in the *forward* direction , i.e., from 1 to T and another network which forms the *backward* component since it works on vectors in the reverse direction, i.e., from time T to 1 we obtain an anti-causal structure also called the *bi-directional* long-short-term memory (BLSTM) neural networks, which capture both *causal* and *anti-causal* recurrence. BLSTMs have been employed with particular success for word spotting tasks. In speech, they have been used primarily to model phonemes [12]. Phonemes have regular, but relatively simple structures that are modeled well by the recurrence in BLSTMs. Word-spotting with BLSTMs has focused on text- or phoneme-sequence-based word specification: the BLSTM first generates a phoneme sequence or lattice from learned phoneme-level models, and the word spotter either scans the phoneme lattice generated with the BLSTMs for the specified words [13], or uses a second-level discriminative classifier that employs features derived from the lattice to detect the words [14]. As such, these methods are two-level classifiers. The BLSTMs require entire recording sequences to be able to do the forward and backward processing, and the secondary classifiers may require further passes through the phoneme sequences or lattices to do the word spotting. Thus these two-level classification strategies might prove to be a deterrent for online processing tasks.

In the work presented here, we employ an RNN with a simple forward-gated structure for example-based *online* word spotting. Unlike previously described approaches, we assume no knowledge of underlying phonetic structure, and no additional information about the language besides the examples of specified words; instead the models are entirely segment based, with each segment modeling an entire word. We do not explicitly find word boundaries. We segment the audio into short, equal-length segments and assign full-word labels to each segment depending on some criteria that we describe later in this paper. Similarly, for online operability during the testing phase, we do not assume the knowledge of word boundaries. Instead we segment the audio into short segments, and classify the individual segments. This approach has inherent problems during training: for example, for training the network, we dont know if it is necessary to capture the temporal sequence of entire words for spotting them, or whether it is sufficient to model words in sections. Furthermore, it is unclear how sensitive the classifier will be to *annotation* that is, exactly how accurately the boundaries of training instances of words must be known in order to train models for them. Even if we use segments that span entire words, we need to investigate whether a simple forward-gated structure will be an effective model for the temporal structure in words (which can be significantly more complex than phonemes). It is also not clear, within this framework, whether a causal formulation could be sufficient for the overall classification. If we automatically block the recording into segments of equal length to enable online processing, we have to contend with the fact that words may cross the segment boundaries in an unpredictable manner; under such circumstances, the effectiveness of our network could be compromised. We investigate these issues in this paper, and show that the block-processing mechanism we propose with our simple network is an effective strategy for word spotting. We find that the network is able to capture the temporal structure of the words well, despite the issues mentioned above. Moreover, it is also remarkably robust to variations introduced by segmentation and imprecision in marking the boundaries of training instances. A much better performance in terms of false alarm rates as measured by the mean time between false alarms (MTBFA) is obtained, which is significantly superior to that obtained with HMMs under similar settings. Furthermore, we also show that the gated RNNs are more robust to noise as compared to the HMM models.

2. GATED RECURRENT NEURAL NETWORK

An RNN has a simple recurrence structure intended to capture temporal structure. However, as mentioned earlier, modeling or learning temporal structure through direct recurrence between hidden units in an RNN is associated with a computational problem: to analyze a time series of T vectors, the RNN has T "columns", one for each time instant. The inputs to the hidden units at any time are derived directly from the units at the previous instance. The problem with this simple recurrence is that the relationship of error at time t to the input at time $t-n$ vanishes quickly with increasing n – this is the well known "vanishing-gradient" (or exploding-gradient) problem [10]. Gated RNNs solve the vanishing gradient problem mentioned above by carrying temporal recurrence through gates [11]. Each gate has a binary value for allowing/disallowing inputs to it for factoring into the output at the current time instant. When gradients become too small, gates simply disallow the inputs to be continued into the recurrence. Figure 1A illustrates a gated RNN. Each gated neuron consists of four components. The architecture is centered around the main "memory" component or the *CEC*. The excitation of the CEC is gated by the "input" and "forget" gates, while its output is gated by the "output" gate. The recurrence relations are given below. In the following equations we use the generic notation: to represent the input to a gate of type Z, where Z could be i (for input), f for (forget), o (for output), or c for inputs to the CEC.

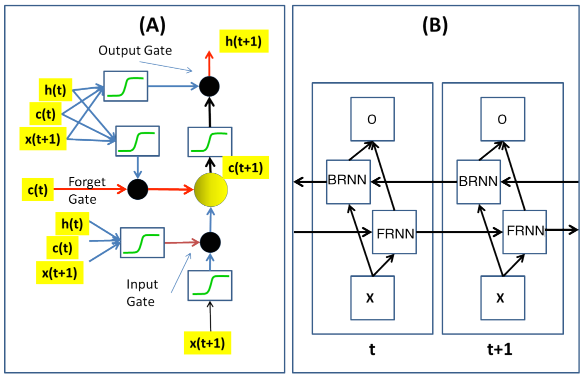

Fig. 1. (A) A gated RNN memory neuron. (B) Bi-directional network. FRNNs are the units in the forward network, while BRNNs are the units in the backward network.

$$\mathbf{I}_Z(t) = \mathbf{W}_Z^x \mathbf{x}(t) + \mathbf{W}_Z^h \mathbf{h}(t-1) \tag{1}$$

$\mathbf{x}(t)$ is the input vector at time t. $\mathbf{W}_Z^x$ represents a matrix of weights from input vector x to the set of gates of type Z. $\mathbf{h}(t)$ is the vector of *gated outputs* from all network units at time t, and $\mathbf{W}_Z^h$ is a matrix of weights from the *gated* outputs to the gates of type Z. We denote the vector of CEC outputs at t as $\mathbf{c}(t)$.

The input and forget gates at t take on values that depend on CEC outputs and gated outputs at $t-1$ as follows:

$$\mathbf{i}(t) = \sigma(\mathbf{I}_i(t) + \mathbf{W}_i^c \mathbf{c}(t-1) + \mathbf{b}_i) \tag{2}$$
$$\mathbf{f}(t) = \sigma(\mathbf{I}_f(t) + \mathbf{W}_f^c \mathbf{c}(t-1) + \mathbf{b}_f) \tag{3}$$

Here $\mathbf{i}(t)$ represents the vector of input gate values at t and $\mathbf{f}(t)$ represents the vector of forget gate values. The σ denotes a sigmoid which ensures the output of the gate is in the range $[0, 1]$. $\mathbf{W}_i^c$ and $\mathbf{W}_f^c$ are diagonal peephole weight matrices from CECs to input and forget gates respectively, and $\mathbf{b}_i$ and $\mathbf{b}_f$ are the biases at all input and forget gates, respectively.

The input and forget gates govern the inputs to the CECs. The CEC output $\mathbf{c}(t)$ at any time also affects the output gates $\mathbf{o}(t)$ at that time, which then gate the CEC output itself to generate the gated output:

$$\mathbf{c}(t) = \mathbf{f}(t) \odot \mathbf{c}(t-1) + \mathbf{i}(t) \odot \tanh(\mathbf{I}_c(t) + \mathbf{b}_c) \tag{4}$$
$$\mathbf{o}(t) = \sigma(\mathbf{I}_o(t) + \mathbf{W}_o^c \mathbf{c}(t) + \mathbf{b}_o) \tag{5}$$
$$\mathbf{h}(t) = \mathbf{o}(t) \odot \tanh(\mathbf{c}(t)) \tag{6}$$

Here $\odot$ represents component-wise multiplication. Thus all three gates (2, (3) and (5) work together to influence what is stored as *memory* in the cell at time t denoted as $\mathbf{c}(t)$ and how this memory influences the final gated outputs $\mathbf{h}(t)$.

These gated neuronal units can form one or more layers in an RNN. Their outputs then feed into a final output layer.

In a *bi-directional* gated RNN [9], also shown in Figure 1B, such gated units are utilized to capture both forward and backward recurrences. The two recurrences are independent of one another, but the gated output values of both networks combine to contribute to the final output of the network.

3. INCREMENTAL EXAMPLE-BASED WORD SPOTTING

In example-based word spotting, we must distinguish between the target word(s), and everything else. In addition, the word spotting must be performed on blocked or segmented audio for incremental processing. Traditional bidirectional gated networks do not require explicit segmentation of each label for training. They can automatically determine label boundaries during the training process by using a CTC like algorithm [7]. Our background model on the other hand, is a catch-all for all sounds that are not keywords and therefore, the model doesn't have a distinctive structure, that can be utilized or learnt effectively for segmentation. For this reason, in our case training from unsegmented data and allowing any algorithm to do automatic segmentation is not expected to be as effective. We therefore need to pre-segment the data for training an online word spotting network. Our approach is as follows: We employ the gated RNN described above in a forward-only configuration, with multiple output units. Each output unit represents a binary encoding of the presence (or absence) of one of the target words. One additional unit encodes the background. We train models from segmented audio. We assign relevant word labels to segments within the target words, and call everything else (including words that are not target words) as "'background"'. We use a *uniform* segmentation, i.e., we segment the audio uniformly into segments of T seconds with an overlap of $T/2$ seconds. If half or more than half the segment under consideration contains a portion of a target word, it is labeled as containing the word, otherwise it is labeled as background. Clearly, words may span multiple segments in this scenario. In such a case, the history of the word is carried across segments by the CEC of the gated RNN. Training is performed through back propagation to minimize the Kullback-Leibler (KL) divergence between the labels output by the network and the vector of true labels for the segments. Detection of the target word is performed simply by classifying each segment into either one of the target words, or the background. Additional details are presented in the experimental section.

As a comparator, in the experimental section below we provide a comparison to the *oracle* segmentation scenario, wherein the precise time-stamps within which the target words occur are assumed to be known. Thus, segments of audio corresponding to the word were provided as positive instances of the target word and segments of audio corresponding to the background were presented as instances of background. In addition we compare the pros and cons of using a forward only model to a bi-directional model.

4. EXPERIMENTS

Experiments were carried out on the TIMIT database. For our experiments, we selected the six most frequently occurring words in the corpus: *greasy*, *oily*, *water*, *carry*, *dark* and *wash*. These words were selected so as to have a comparison of our model with the results in [2]. All other words were considered to be part of the background. The training data used was the TIMIT training set, comprising 3696 sentences spoken by 462 speakers across eight dialects. We used two-thirds of the original TIMIT train set for training the network and the remaining one-third was used as the cross-validation set. The test set was the same as the original TIMIT test set comprising 1344 utterances from 168 speakers. There were a total of about 460 training instances of each keyword with 300 in the training set and about 160 in the cross validation set. The test set had 168 instances of each keyword. All the other words made up the background garbage model.

For all experiments, the speech signal was transformed into a sequence of 39- dimensional feature vectors consisting of 13-dimensional MFCC vectors appended with their Δ and $\Delta\Delta$ terms, computed from analysis windows of 25ms, with a 50% overlap between frames. In order to evaluate the ability of the units to capture the detailed temporal structure in words, we used a relatively small network with 26 hidden units, and as many output neurons as the number of output labels, that is, the number of keywords + one for background.

The importance of the segmentation length: In the first experiment we evaluated the effect of segment length on performance, since this is an important parameter when considering the buffer size in an online speech recognition setting. Simply utilizing the gated-network to perform segmentation intrinsically resulted in extremely poor performance. Thus, we explored training the network with four different segment lengths. The segment lengths that were used were 156 ms, 306 ms, 500 ms and 1s, since the average word length across *all* words in the train set was about 306 ms. Fig. 2, lists the recall (in %) and MTBFA (in ms) for different segment lengths as well as for an oracle segmentation and HMM models. With respect to gated RNN models, model A corresponds to a forward model while model B is the bidirectional *anti-causal* model. In case of the HMM, model A has an extended dictionary with all other words mapped to garbage, while in model B, the dictionary only contains the keywords and one alternate pronunciation for garbage. The HMM models used were 3-state context dependent phoneme models with 3000 senones with 8 gaussians per state and a diagonal covariance matrix. As can be seen in Fig.2, the plots peak for the oracle model, both in terms of MTBFA and recall. Thus, as expected, the gated RNN trained on oracle segmented data performs much better than uniform segment models as well as the HMM models. This indicates that selecting a segment length which is the average size of the keywords might be a better choice as compared to selecting a word size that spans an average length of all words in the training set. If we compare the uniform segment models, we see that the best performance overall, in terms of MTBFA and recall rates is given by the 500 ms model across all keywords, even though the 1s models performs better than the 500 ms model for certain keywords, it has a greater variance across keywords and thus we believe that the 500 ms is the best in terms of the uniform segment length. Moreover, the MTBFA rates that are achieved by this model, which requires no segmentation and no word boundary information, is significantly better than MTBFA rates achieved by an HMM. However, the 500 ms uniform segment model does lose out on recall by about 8% as compared to HMM model B, which is not a very heavy penalty to pay for the significantly lower false alarm rates or higher MTBFA, since a missed detection is not as expensive as a false alarm for this task.

The impact of bi-directionality: In this set of experiments, we wanted to compare the performance of a bidirectional model as compared to a forward model, to see what is the optimal performance one can obtain from a gated structure. Fig 2 compares the two. We see that across all segment lengths, the bidirectional network has some performance improvements as compared to the forward only model. Both models seem to have similar recall rates across segment lengths, however, the bidirectional model gains slightly in terms of precision which in turn leads to higher MTBFA. This shows us that by adding in the extra backward pass, one can increase the precision, without having too much of a fallback on the recall rates. However, under this scenario we would no longer have a purely causal system and there would always be some lag between the input and the decision of the network.

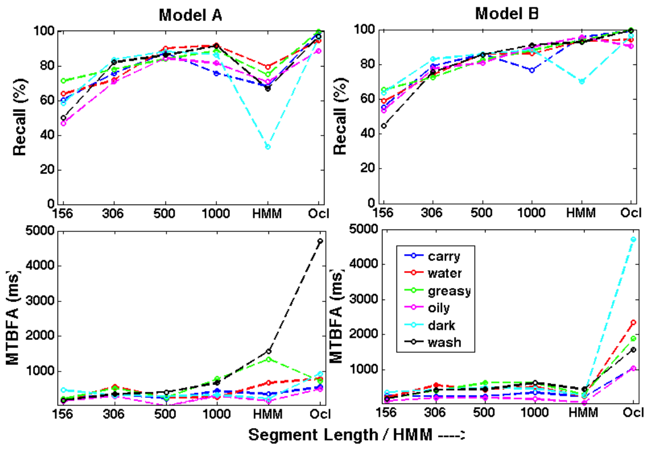

Fig. 2. Comparison of two models trained with 4 segment lengths *(156, 306, 500 and 1000 ms)*. Also shown are performance obtained with oracle segmentation (Ocl) and HMMs. *Model A*- Forward-only gated RNN. *Model B* - Bidirectional gated RNN.

The effect of training size: To learn how the network performance scales with an increase in the training set size, we tested for different sizes of the training set (in terms of number of keywords used for training). The total number of keywords in the training set was about 300 keywords of each type from a total of 31942 words with 30130 words comprising the garbage words. The results as are shown in Fig. 3 were obtained by varying the number of instances of keywords from 50 keywords of each type, which included a total of 4915 words of which 4635 were garbage words, to 300 instances of keywords. As can be seen, the main effect of increasing the training set size is on the recall rates which increases linearly in most cases with respect to the increase in training set size across all keywords.

Noise robustness: Table 1 compares the performance of the best gated RNN model with the HMM for two noise levels 5db and 10db. the noise was mixed noise of various kinds which included other speech and music. The models were trained on this noisy speech. We see that we get significantly higher recall rates as compared to HMM models, without much loss in precision as seen by the slightly lower MTBFA rates for gated RNN model. This clearly shows us that the

Table 1. Comparison of noise robustness of gated RNN and HMM for two noise levels 5db and 10db of mixed noise Legend: *Carry (C)*,*Water (W)*,*Greasy (G)* and *Oily (O) Dark (D)* and *Wash (Wa)*

Noise/	Key	Bidirectional-gated RNN		HMM	
		Rec	MTBFA(*s*)	Rec	MTBFA(*s*)
5db	C	0.828	555.05	0.272	1179.48
	W	0.682	786.33	0.194	1572.64
	G	0.779	1048.44	0.315	1347.98
	O	0.762	124.157	0.417	131.05
	D	0.789	496.63	0.169	2358.96
	Wa	0.809	629.06	0.178	9435.84
10db	C	0.887	314.53	0.432	725.83
	W	0.830	629.06	0.329	857.80
	G	0.851	3145.33	0.524	673.989
	O	0.762	337.0	0.583	88.18
	D	0.865	589.75	0.257	2358.96
	Wa	0.869	1048.44	0.327	2358.96

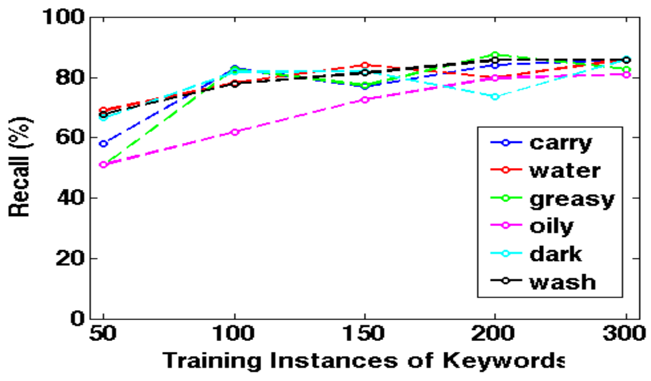

Fig. 3. Effect of increasing number of training instances of keywords for 6 different keywords.

gated RNN models are much more robust to noise as compared to the HMM models, which might be attributed to the sequence structure that is learnt by these gated memory networks.

The importance of the garbage model: To have a robust word spotter, it is of paramount importance that the background be modeled well. We can model the garbage in two ways: using a *generic* model, which simply encompasses all alternate words, or through a more *structured* model which explicitly models some alternate words, in addition to the generic background. Table 2 compares these two methods. Here, for each of our six target words we trained two classifiers, one a binary word-vs-background classifier, and a second multi-class classifier, where the word was the primary target and all other words (the remaining target words) were grouped with the background. Comparing the recall and MTBFA rates in Table 2 column 1 to recall rates in column 2 corresponding to the generic garbage model, we see that across the different keywords, there is no set pattern as to which model performs better. However, in most cases it has comparable recall and false alarm rates. Thus, choosing which model to use would depend more on our application, since the structured model would afford us more flexibility to change keywords on the fly as it has one separate model for each keyword. However, for the generic model, a change of keywords would mean retraining the entire model.

Table 2. Comparison of structured garbage model for the different keywords using 500 ms uniform segmented training data.. Legend: *Carry (C)*,*Water (W)*,*Greasy (G)* and *Oily (O) Dark (D)* and *Wash (Wa)*

Key	Structured Model		Generic Model	
	Rec	MTBFA(*s*)	Rec	MTBFA(*s*)
C	0.853	277.53	0.858	248.31
W	0.819	725.85	0.86	428.90
G	0.80	496.63	0.827	629.06
O	0.846	188.72	0.81	209.69
D	0.860	362.92	0.86	496.63
Wa	0.876	269.6	0.857	449.33

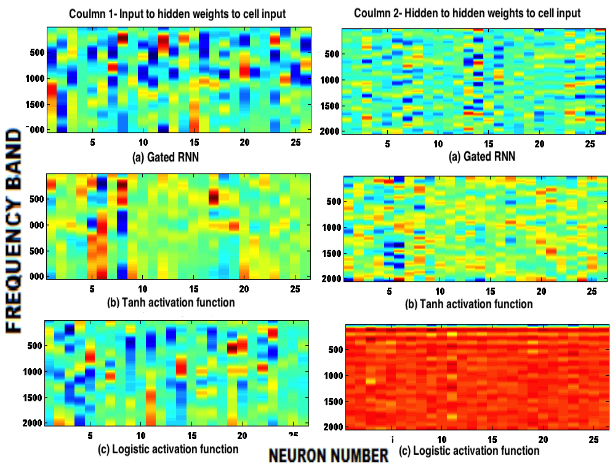

Fig. 4. Plots of the input to hidden weights and hidden to hidden weights to the cell input for networks with gated RNN blocks, tanh activation function, and logistic activation function

The impact of memory gating: The gated RNN introduces many additional parameters compared to a simple recurrent network, mainly due to the weights associated with all the gates. The benefit of these gates and the memory structure they protect has generally been taken for granted; here we confirm the advantage. Figure 5 compares a conventional RNN with two different types of hidden activations, a tanh and sigmoid activation function with a gated RNN. The gated RNN clearly outperforms other models.

In addition, Figure 4 shows learned weight matrices for different RNNs' activation functions, and compares them with the gated RNN. Since neurons in a neural network effectively act as matched filters it is interesting to note *what* they have learned as the basic patterns in their weight matrices. We have performed an IDCT to invert the patterns from the quefrency domain to the frequency domain. We note that for the gated RNN models, the neurons strongly accept or reject frequencies, presumably those that uniquely identify the target words. The conventional RNN activations, on the

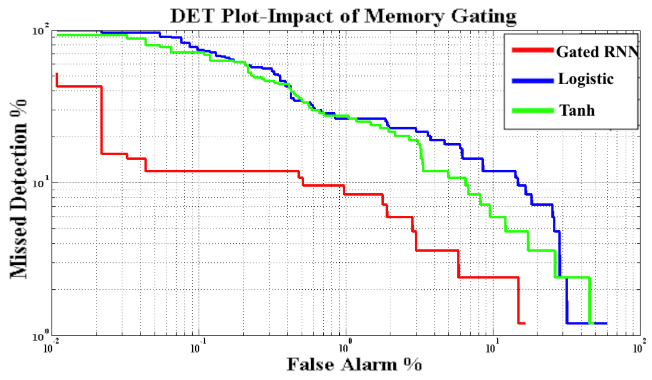

Fig. 5. DET Plots for different hidden neurons, tanh, gated RNN and logistic, for the word *water*

other hand, show a more uniform structure, demonstrating a reduced ability to learn discriminative features. The distinctions are particularly stark for the weights on the recurrent connections. The gating functions appear to introduce a key discriminative ability, and are in fact necessary for good accuracy and to prevent the recurrent (hidden-to-hidden) connections from getting saturated, even for memories that must span only a fraction of a second.

5. DISCUSSION

This paper proposes and evaluates a simple *sectioned* gated RNN architecture for example-based word spotting. Unlike [3] and [13] solutions that also use gated memory networks, the proposed approach does not require the use of a secondary algorithm like the CTC algorithm to carry out the sequence labeling over the classified phone labels. Instead, in our approach we directly classify the input data into either target or background words in a left-to-right online manner. The paper investigates various factors that matter in this process - such as optimal segment or buffer size to consider, appropriate background model, issues relating to size of training set and noise robustness etc. It was found that selecting an appropriate buffer size plays an important role in affecting false alarm and recall rates. We found that the best segment length overall lies in a range corresponding to the average duration of *keywords* in the training set. Modeling the background appropriately was an issue that was also investigated. It was found that across keywords, there wasn't really a clear winner as to which was the appropriate method to model the garbage. The structured garbage model has a slight advantage over more detailed, generic models for detecting specific keywords. However, this difference is not very large, and in systems where simple training is desired, a generic background model should serve the purpose equally well. We note that the gated RNN, even when applied in a blocked fashion, does provide improved recall and false alarm rates. The effect on the network performance with an increase in training set size on the other hand shows a linear increase in the recall rates. In addition, the biggest advantage the gated RNN network seems to have as compared to HMM's is their robustness to noise. In the blocked processing that we introduce, words may span segment boundaries. We observe that the gating mitigates many of the ill-effects of this gross representation as seen by comparing a gated RNN *vs.* a non-gated RNN. Finally, in spite of the very simple model structures used, we achieve good performance, obtaining an effective false alarm rate of less than one per 500 seconds on an average at relatively low missed detection rates. In future we plan to investigate mechanisms for carrying information across segments, minimizing the training examples needed, and improving robustness to noise.

6. REFERENCES

[1] R.C. Rose and D.B. Paul, "A Hidden Markov Model Based Keyword Recognition System," *ICASSP*,1990.

[2] T. Ezzat, T. Poggio, "Discriminative Word-Spotting Using Ordered Spectro-Temporal Patch Features," *SAPA workshop*, 2008.

[3] S.Fernandez, A. Graves, J. Schmidhuber," An Application of Recurrent Neural Networks to Discriminative Keyword Spotting," *In: Proc. ICANN*, Porto, Portugal, pp. 220 - 229. 2007.

[4] T. J. Hazen, W. Shen and C. White, "Query-by-example Spoken Term Detection Using Phonetic Posteriogram Templates," *Proc. ASRU*, 2009.

[5] A. Jansen, and P. Niyogi, "Point process models for spotting keywords in continuous speech," *IEEE Trans. on Audio, Speech, and Language Proc., 17, no. 8* , pp. 1457-1470, 2009.

[6] T. N. Sainath, A. Mohamed, B. Kingsbury, B. Ramabhadran, "Deep Convolutional Neural Networks for LVCSR", *in ICASSP* , 2013.

[7] A. Graves, A. Mohamed, G. Hinton, "Speech Recognition with Deep Recurrent Neural Networks," *in ICASSP*, Canada, 2013.

[8] G. Dahl, M. Ranzato, A. Mohamed, G. Hinton, "Phone Recognition with the Mean-Covariance Restricted Boltzmann Machine, " *In Advances in Neural Info. Proc. Systems 23*, pp. 469-477, 2010.

[9] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans on Signal Proc., 45, no. 11* pp.2673-2681, 1997.

[10] S. Hochreiter, Y. Bengio, P. Frasconi and J. Schmidhuber, "Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies, " *in A Field Guide to Dynamical Recurrent Neural Networks*, IEEE Press, 2001.

[11] F. A. Gers, "Long Short-Term Memory in Recurrent Neural Networks", PhD thesis, Dept. of Computer Science, Swiss Federal Institute of Technology, Lausanne, EPFL, Switzerland, 2001.

[12] A. Graves, S. Fernndez, F. Gomez, J. Schmidhuber, "Connectionist Temporal Classification: Labeling Unsegmented Sequence Data with Recurrent Neural Networks,"*ICML*, USA, pp. 369-376, 2006

[13] M. Wollmer, F. Eyben, A. Graves, B. Schuller and G. Rigoll, "Improving Keyword Spotting with a Tandem BLSTM-DBN Architecture," *in Non-Linear Speech Processing, J. Sole-Casals and V. Zaiats (Eds.), LNAI 5933*, pp. 68-75, Springer Heidelberg, 2010

[14] Y. Sun, T. Bosch, and L. Boves, "Hybrid HMM/BLSTM-RNN for Robust Speech Recognition," *In Proc. of the 13th International Conference on Text, Speech and Dialogue* pp. 400-407. Springer-Verlag. September 2010