International Journal on Applications in Engineering and Technology
Volume 4: Issue 2: April 2018, pp 1 - 6. www.aetsjournal.com ISSN (Online) : 2455 - 0523

SMALL – FOOT PRINT KEYWORD SPOTTING USING DEEP NEURAL NETWORKS

DHATCHAYINI.B, R.SIVAPRAKASH , R.VETRIVENTHAN, SENTHIL KUMARAN

Abstract— Our application requires a keyword spotting system with a small memory footprint, low computational cost, and high precision. To meet these requirements, we propose a simple approach based on deep neural networks. A deep neural network is trained to directly predict the keyword(s) or subword units of the keyword(s) followed by a posterior handling method producing a final confidence score. Keyword recognition results achieve 45% relative improvement with respect to a competitive Hidden Markov Model-based system, while performance in the presence of babble noise shows 39% relative improvement.

Keywords— Keyword Spotting (KWS) , DNN based KWS.

I. INTRODUCTION

Thanks to the rapid development of smartphones and tablets, inter acting with technology using voice is becoming commonplace. For example, Google offers the ability to search by voice [1] on Android devices and Apple's iOS devices are equipped with a conversational assistant named Siri. These products allow a user to tap a device and then speak a query or a command.

We are interested in enabling users to have a fully hands-free experience by developing a system that listens continuously for specific keywords to initiate voice input. This could be especially useful in situations like driving. The proposed system must be highly accurate, low-latency, small-footprint, and run in computationally constrained environments such as modern mobile devices. Running the system on the device avoids latency and power implications with connecting to the server for recognition.

Keyword Spotting (KWS) aims at detecting predefined keywords in an audio stream, and it is a potential technique to provide the desired hands-free interface. There is an extensive literature in KWS, although most of the proposed methods are not suitable for low-latency applications in computationally constrained environ- ments. For example, several KWS systems [2, 3, 4] assume offline processing of the audio using large vocabulary continuous speech recognition systems (LVCSR) to generate rich lattices. In this case, their task focuses on efficient indexing and search for keywords in the lattices. These systems are often used to search large databases of audio content. We focus instead on detecting keywords in the audio stream without any latency.

A commonly used technique for keyword spotting is the Keyword/Filler Hidden Markov Model (HMM) [5, 6, 7, 8, 9]. Despite being initially proposed over two decades ago, it remains highly competitive. In this generative approach, an HMM model is trained for each keyword, and a filler model HMM is trained from the non- keyword segments of the speech signal (fillers). At runtime, these systems require Viterbi decoding, which can be computationally expensive depending on the HMM topology. Other recent work explores discriminative models for keyword spotting based on large- margin formulation [10, 11] or recurrent neural networks [12, 13]. These systems show improvement over the HMM approach but re- quire processing of the entire utterance to find the optimal keyword region or take information from a long time span to predict the entire keyword, increasing detection latency.

We propose a simple discriminative KWS approach based on deep neural networks that is appropriate for mobile devices. We refer to it as Deep KWS . A deep neural network is trained to directly predict the keyword(s) or subword units of the keyword(s) followed by a posterior handling method producing a final confidence score. In contrast with the HMM approach, this system does not require a sequence search algorithm (decoding), leading to a significantly simpler implementation, reduced runtime computation, and smaller memory footprint. It also makes a decision every 10 ms, minimizing latency. We show that the Deep KWS system outperforms a standard HMM based system on both clean and noisy test sets, even when a smaller amount of data is used for training.

We describe our DNN based KWS framework in Section 2, and the baseline HMM based KWS system in Section 3. The experimental setup, results and some discussion follow in Section 4. Section 5 closes with the conclusions.

II. DEEP KWS SYSTEM

The proposed Deep KWS framework is illustrated in Figure 1. The framework consists of three major components: (i) a feature extrac- tion module, (ii) a deep neural network, and (iii) a posterior handling module. The feature extraction module (i) performs voice-activity detection and generates a vector of features every frame (10 ms). These features are stacked using the left and right context to create a larger vector,

Dhatchayini.B, Master of Computer Applications , Meenaakshi Ramasamy Engineering College , Thathanur , Ariyalur Dt , Tamil Nadu .

R.Sivaprakash MCA.M.Phil. Assistant professor/MCA , Meenaakshi Ramasamy Engineering College , Thathanur , Ariyalur Dt , Tamil Nadu .

Mr.R.Vetriventhan , BE , MTech , MISTE , Head of the department , Department of MCA , Meenaakshi Ramasamy Engineering College , Thathanur , Ariyalur Dt , Tamil Nadu .

Mr.Senthil Kumaran , ME Phd , Managing Director , Meenaakshi Ramasamy Engineering College , Thathanur , Ariyalur Dt , Tamil Nadu.

which is fed as input to the DNN (Section 2.1). We train a DNN (ii) to predict posterior probabilities for each out- put label from the stacked features. These labels can correspond to entire words or sub-words for the keywords (Section 2.2). Finally, a simple posterior handling module (iii) combines the label posteri- ors produced every frame into a confidence score used for detection (Section 2.3).

In the example of Figure 1, the audio contains the key-phrase "okay google". The DNN in this case only has 3 output labels: "okay", "google", and "filler", and it generates frame-level poste- rior scores shown in (iii). The posterior handling module combines these scores to provide a final confidence score for that window.

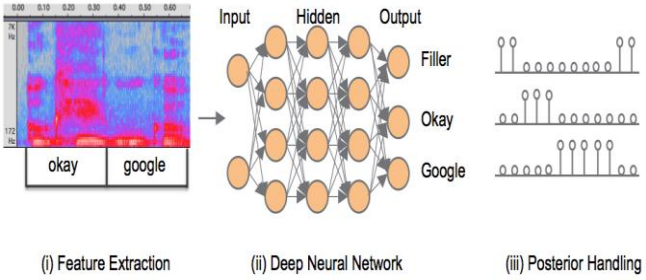

Figure 1. Framework of Deep KWS system, components from left to right: (i) Feature Extraction (ii) Deep Neural Network (iii) Posterior Handling

1) Feature Extraction

The feature extraction module is common to our proposed Deep KWS system and the baseline HMM system.

To reduce computation, we use a voice-activity detection system and only run the KWS algorithm in voice regions. The voice-activity detector, described in [14], uses 13-dimensional PLP features and their deltas and double-deltas as input to a 30-component diagonal covariance GMM trained, which generates speech and non-speech posteriors at every frame. This is followed by a hand-tuned state machine (SM), which performs temporal smoothing by identifying regions where many frame speech posteriors exceed a threshold.

For the speech regions, we generate acoustic features based on 40-dimensional log-filterbank energies computed every 10 ms over a window of 25 ms. Contiguous frames are stacked to add sufficient left and right context. The input window is asymmetric since each additional frame of future context adds 10 ms of latency to the sys- tem. For our Deep KWS system, we use 10 future frames and 30 frames in the past. For the HMM baseline system we use 5 future frames and 10 frames in the past, as this provided the best trade-off between accuracy, latency, and computation [15].

2) Deep Neural Network

The deep neural network model is a standard feed-forward fully connected neural network with k hidden layers and n hidden nodes per layer, each computing a non-linear function of the weighted sum of the output of the previous layer. The last layer has a softmax which outputs an estimate of the posterior of each output label. For the hidden layers, we have experimented with conventional logistic and rectified linear unit (ReLU) functions [16], and consistently found that ReLU outperforms logistic on our development set, while reducing computation. We present results with ReLU activations only.

The size of the network is also dictated by the number of output labels. In the following sub-sections we describe in detail the label generation and training for our neural network. We also describe a learning technique that further improves the KWS performance.

Labeling. For our baseline HMM system, as in previous work [8, 9, 17] the labels in the output layer of the neural network are context dependent HMM states. More specifically the baseline system uses 2002 context dependent states selected as described in .

For the proposed Deep KWS , the labels can represent entire words or sub-word units in the keyword/key-phrase. We report results with full word labels, as these outperform sub-word units. These labels are generated at training time via forced alignment using our 50M parameter LVCSR system. Using entire word labels as output for the network, instead of the HMM states, has sev- eral advantages: (i) smaller inventory of output labels reduces the number of neural network parameters in the last layer, which is com-putationally expensive (ii) a simple posterior handling method can be used to make a decision (as explained in Section 2.3), (iii) whole word models achieve better performance, assuming the training data is adequate for each word label considered.

Training. Suppose p_{ij} is the neural network posterior for the i^{th} label and the j^{th} frame x_j (see Section 2.1), where i takes values between 0, 1, ..., $n - 1$, with n the number of total labels and 0 the label for non-keyword. The weights and biases of the deep neural network, θ, are estimated by maximizing the cross-entropy train- ing criterion over the labeled training data $\{x_j, i_j\}_j$ (previous para- graph).

$$F(\theta) = \sum_j \log p_{i_j,j} \qquad (1)$$

The optimization is performed with the software framework DistBe- lief [19, 20] that supports distributed computation on multiple CPUs in deep neural networks. We use asynchronous stochastic gradient descent with an exponential decay for the learning rate.

3) Transfer learning.

Transfer learning refers to the situation where (some of) the network parameters are initialized with the corre-sponding parameters of an existing network, and are not trained from scratch [21, 22]. Here, we use a deep neural network for speech recognition with suitable topology to initialize the hidden layers of the network. All layers are updated in training. Trans- fer learning has the potential advantage that the hidden layers can learn a better and more

robust feature representation by exploiting larger amounts of data and avoiding bad local optima [21]. In our experiments we find this to be the case.

4) Posterior Handling

The DNN explained in Section 2.2 produces frame-based label pos- teriors. In this section we discuss our proposed simple, yet effective, approach to combine DNN posteriors into keyword/key-phrase con- fidence scores. A decision then will be made if the confidence ex- ceeds some predefined threshold. We describe the confidence com- putation assuming a single keyword. However, it can be easily mod- ified to detect multiple keywords simultaneously.

Posterior smoothing. Raw posteriors from the neural network are noisy, so we smooth the posteriors over a fixed time window of size w_{smooth}. Suppose p^0_{ij} is the smoothed posterior of p_{ij} (Eq. 1). The smoothing is done with the following formula:

$$p^0_{ij} = \frac{1}{j - h_{smooth} + 1} \sum_{k=h_{smooth}}^{j} p_{ik} \quad (2)$$

where $h_{smooth} = \max\{1, j - w_{smooth} + 1\}$ is the index of the first frame within the smoothing window.

5) Confidence.

The confidence score at j^{th} frame is computed within a sliding window of size w_{max}, as follows

$$confidence = \max_{t} \prod_{i=1}^{n-1} \max_{h_{max} \le k \le j} p^0_{ik} \quad (3)$$

where p^0 is the smoothed state posterior in Eq. (2), $h_{max} = \max\{1, j - w_{max} + 1\}$ is the index of the first frame within the slid- ing window. We use $w_{smooth} = 30$ frames, and $w_{max} = 100$, as this gave best performance on the dev set; however the performance was not very sensitive to the window sizes. Eq. (3) does not enforce the order of the label sequence, however the stacked frames fed as input to the neural network help encode contextual information.

III. BASELINE HMM KWS SYSTEM

We implement a standard Keyword-Filler Hidden Markov Model as our baseline. The basic idea is to create a HMM for the keyword and a HMM to represent all non-keyword segments of the speech signal (filler model). There are several choices for the filler model, from fully connected phonetic units [6] to a full LVCSR system where the lexicon excludes the keyword [23]. Obviously, the latter approach yields a better filler model, however it requires higher computational cost at runtime, and significantly larger memory footprint. Given the constraints of our application, we implemented a triphone-based HMM model as filler. In contrast to previous work [6, 23], our im- plementation uses a Deep Neural Network to compute the HMM state densities.

The Keyword-Filler HMM topology is shown in Figure 2. Keyword detection is achieved by running Viterbi decoding with this topology and checking if the best path passes through the Keyword HMM or not. The trade-off between false alarms (a keyword is not present but the KWS system gives a positive decision) and false re- jects (a keyword is present but the KWS system gives a negative decision) is controlled by the transition probability between key- word and filler models. High transition probability leads to high false alarm rate and vice versa.

An important advantage of the Keyword-Filler model is that it does not require keyword-specific data at training time. It simply learns a generative model for all triphone HMM states through like- lihood maximization on general speech data. Knowledge of the key- word can be introduced only at runtime, by specifying the keyword in the decoder graph. However, if keyword-specific data is avail- able for training, one can improve system performance using trans- fer learning (Section 2.2), i.e., by initializing the acoustic model net- work with a network trained on the general speech data and then continue training it using the keyword-specific data.

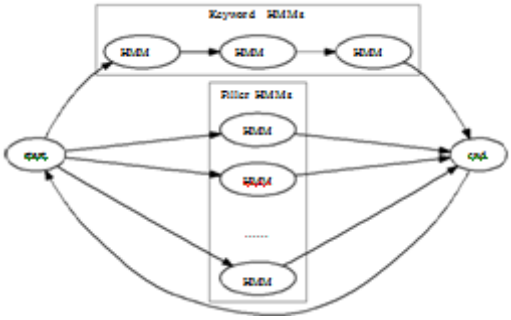

Figure 2. HMM topology for KWS system, which consists of a key- word model and a triphone filler model

IV. EXPERIMENTAL RESULTS

Experiments are performed on a data set which combines real voice search queries as negative examples and phrases including the keywords, sometimes followed by queries, as positive examples. A full list of the keywords evaluated is shown in Table 1. We train a sep- arate Deep KWS and build a separate Keyword-Filler HMM KWS system for each keyphrase. Results are presented in the form of a modified receiver operating characteristic (ROC) curves, where we replace true positive rate with the false reject rate on Y-axis. Lower curves are better. The ROC for the baseline system is obtained by sweeping the transition probability for the Keyword HMM path in Figure 2. For the Deep KWS system, the ROC is obtained by sweep- ing the confidence threshold. We generate a curve for each keyword and average the curves vertically (at fixed FA rates) over all keywords tested.

TABLE 1 - KEYWORDS USED IN EVALUATION

answer call	dismiss alarm
go back	ok google
read aloud	record a video
reject call	show more commands
snooze alarm	take a picture

We compare the Deep KWS system and the HMM system with different size neural networks (Section 4.3), evaluate the effect of transfer learning for both systems (Section 4.2), and show performance changes in the presence of babble noise (Section 4.4).

1) Data

We use two sets of training data. The first set is a general speech corpus, which consists of 3,000 hours of manually transcribed utter- ances (referred to as VS data). The second set is a keyword specific data (referred to as KW data), which included around 2.3K train- ing examples for each keyword, and 133K negative examples, com- prised of anonymized voice search queries or other short phrases. For the keyword "okay google", 40K positive examples were avail- able for training.

The evaluation set contains roughly 1K positive examples for each keyword and 70K negative examples, representing 1.4% of pos- itive to negative ratio, to match expected application usage. Again, for keyword "okay google" we used instead 2.2K positive examples. The noisy test set was generated by adding babble noise to this test set with a 10db Signal to Noise Ratio (SNR). Finally, we use a sim- ilar size non-overlapping set of positive and negative examples as development set to tune decoder parameters and DNN input window size parameters.

2) Results

We first evaluate the performance of the smaller neural network trained for the baseline HMM and the Deep KWS systems. Both systems used the frontend described in 2.1. They both used a network with 3 hidden layers and 128 hidden nodes per layer with ReLU non-linearity. However, the number of parameters for both networks is not identical. The DNN acoustic model used for the baseline HMM system uses an input window size of 10 left frames and 5 right frames, and outputs 2,002 HMM states, resulting in around 373M parameters. The Deep KWS uses instead a 30 left frames and 10 right frames, but only produces word labels reducing the output label inventory to 3 or 4 depending on the key-phrase evaluated. The total number of parameters for Deep KWS is no larger than 244M parameters.

Figure 3 shows the performance for both systems. Baseline 3x128 (VS) refers to the HMM system with a DNN acoustic model trained on the voice search corpus. Baseline 3x128 (VS + KW) is this same system after adapting the DNN acoustic model using keyword specific data. Deep 3x128 (KW) refers to the proposed Deep KWS system trained on keyword specific data. Finally, Deep 3x128 (VS + KW) shows the performance when we initialize the Deep 3x128 KW network with a network trained on VS data as explained in Section 2.2.

It is clear from the results that the proposed Deep KWS out- performs the baseline HMM KWS system even when it is trained with less data and has fewer number of parameters. For example, see Deep 3x128 (KW) vs Baseline 3x128 (VS + KW) in Figure 3. The gains are larger at very low false alarm rate, which is a desir- able operating point for our application. At 0.5% FA rate, Deep

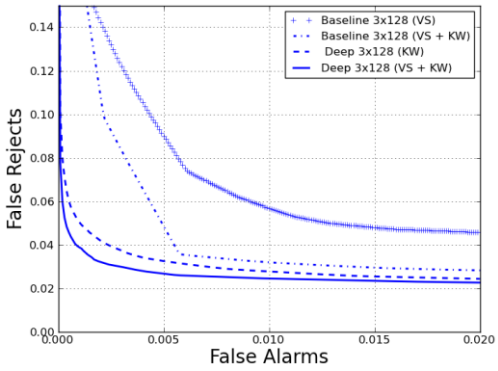

Figure 3. HMM vs. Deep KWS system with 3 hidden layers, 128 hidden nodes neural network

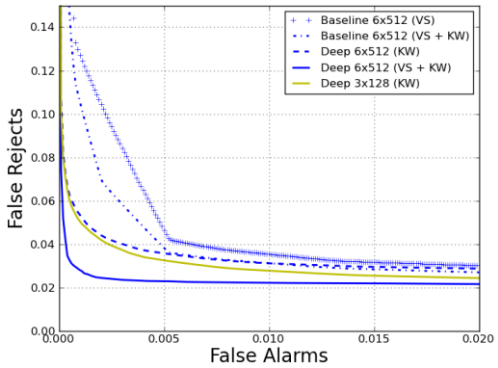

Figure 4. HMM vs. Deep KWS system with 6 hidden layers, 512 hidden nodes neural network 3x128 (VS + KW) system achieves 45% relative improvement with respect to Baseline 3x128 (VS + KW). Training a network on the KW data takes only a couple of hours, while training it on VS + KW takes about a week using our DistBelief framework described in Section 2.2.

3) Model Size

Figure 4 presents the performance when evaluating both systems with a 6x512 network. In this case the number of parameters for the baseline increases to 2.6M while the Deep models reach 2.1M. Deep 6x512 (KW) system, actually performs worse than the smaller 3x128 models, we conjecture this is due to not enough KW data to train the larger number of parameters. However when both systems are trained on VS + KW data, we observe a consistent improvement with respect to their corresponding 3x128 systems. Here again, the Deep

KWS system has superior performance to the baseline.

4) Noise Robustness

We also test the same models on a noisy test set, generated by adding babble noise to the original test set with a 10db SNR. Comparing Baseline 3x128 (VS + KW) in Figure 3 and Figure 5, at 0.5% FA

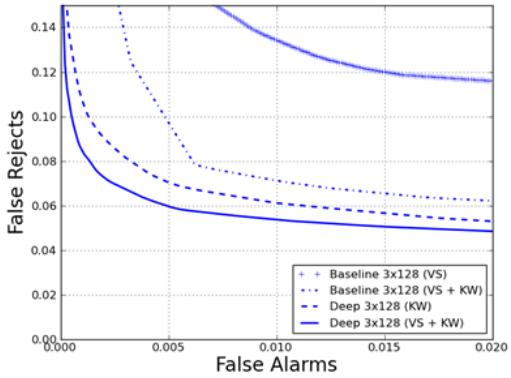

Figure 5. HMM vs. Deep KWS system with 3 hidden layers, 128 hidden nodes neural network on NOISY data

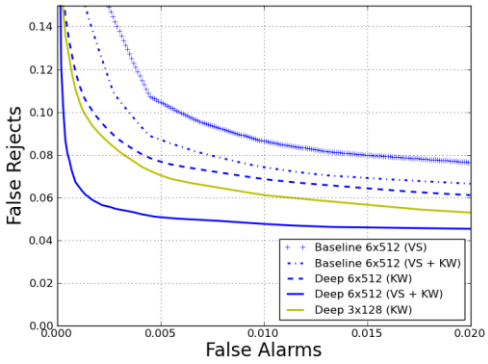

Figure 6. HMM vs. Deep KWS system with 6 hidden layers, 512 hidden nodes neural network on NOISY data rate, the FR rate of the HMM doubles from 5% FR to 10% FR. The Deep KWS system suffers similar degradation. However it achieves 39% relative improvement with respect to the baseline.

V. CONCLUSION

We have presented a new deep neural network based framework for keyword spotting. Experimental results show that the proposed framework outperforms the standard HMM based system on both clean and noisy conditions. We further demonstrate that a Deep KWS model trained with only the KW data yields better search per- formance over the baseline HMM KWS system trained with both KW and VS data. The Deep KWS system also leads to a simpler implementation removing the need for a decoder, reduced runtime computation, and a smaller model, and thus is favored for our em- bedded application.

VI. ACKNOWLEDGEMENTS

We gratefully acknowledge many fruitful discussions with Alexander Gruenstein and other members of the Google Mobile Speech team. We also thank Mark Mao for his help with DistBelief.

References

[1] Johan Schalkwyk, Doug Beeferman, Françoise Beaufays, Bill Byrne, Ciprian Chelba, Mike Cohen, Maryam Kamvar, and Brian Strope, ""Your word is my command": Google search by voice: A case study," in Advances in Speech Recognition, pp. 61–90. Springer, 2010.

[2] David RH Miller, Michael Kleber, Chia-Lin Kao, Owen Kimball, Thomas Colthurst, Stephen A Lowe, Richard M Schwartz, and Herbert Gish, "Rapid and accurate spoken term detection.," in INTERSPEECH, 2007, pp. 314–317.

[3] Siddika Parlak and Murat Saraclar, "Spoken term detection for turkish broadcast news," in Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2008, pp. 5244–5247.

[4] Jonathan Mamou, Bhuvana Ramabhadran, and Olivier Siohan, "Vocabulary independent spoken term detection," in Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2007, pp. 615–622.

[5] J.R. Rohlicek, W. Russell, S. Roukos, and H. Gish, "Continuous hidden Markov modeling for speaker-independent wordspotting," in Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 1990, pp. 627–630.

[6] Richard C Rose and Douglas B Paul, "A hidden Markov model based keyword recognition system," in Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 1990, pp. 129–132.

[7] JG Wilpon, LG Miller, and P Modi, "Improvements and applications for key word recognition using hidden Markov modeling techniques," in Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 1991, pp. 309–312.

[8] Marius-Calin Silaghi and Hervé Bourlard, "Iterative posterior-based keyword spotting without filler models," in Proceedings of the Automatic Speech Recognition and Understanding Work- shop (ASRU). IEEE, 1999, pp. 213–216.

[9] Marius-Calin Silaghi, "Spotting subsequences matching an HMM using the average observation probability criteria with application to keyword spotting.," in Proceedings of the National Conference on Artificial Intelligence. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005, vol. 20, p. 1118.

[10] David Grangier, Joseph Keshet, and Samy Bengio, "Discriminative keyword spotting," Automatic speech and speaker recognition: large margin and kernel methods, pp. 175–194, 2009.

[11] Shima Tabibian, Ahmad Akbari, and Babak Nasersharif, "An evolutionary based discriminative system for keyword spotting," in International Symposium on Artificial Intelligence and Signal Processing (AISP). IEEE, 2011, pp. 83–88.

[12] KP Li, JA Naylor, and ML Rossen, "A whole word recurrent neural network for keyword spotting," in Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 1992, vol. 2, pp. 81–84.

[13] Santiago Fernández, Alex Graves, and Jürgen Schmidhuber, "An application of recurrent neural networks to discriminative keyword spotting," in Artificial Neural Networks–ICANN 2007, pp. 220–229. Springer, 2007.

[14] Thad Hughes and Keir Mierle, "Recurrent neural networks for voice activity detection," in Proceedings of the Interna- tional Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2013, pp. 7378–7382.

[15] Xin Lei, Andrew Senior, Alexander Gruenstein, and Jeffrey Sorensen, "Accurate and compact large vocabulary speech recognition on mobile devices," in INTERSPEECH, Apr. 2013, vol. 1.

[16] Matthew D Zeiler, Marc'Aurelio Ranzato, Rajat Monga, Mark Mao, Ke Yang, Quoc V Le, Patrick Nguyen, Andrew Senior, Vincent Vanhoucke, Jeff Dean, and Geoffrey E Hinton, "On rectified linear units for speech processing," in Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, Canada, Apr. 2013, IEEE, vol. 1.

[17] George E Dahl, Dong Yu, Li Deng, and Alex Acero, "Large vocabulary continuous speech recognition with context-dependent DBN-HMMs," in Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2011, pp. 4688–4691.

[18] N. Jaitly, P. Nguyen, A. Senior, and V. Vanhoucke, "Application of pretrained deep neural networks to large vocabulary speech recognition," in INTERSPEECH, 2012.

[19] Quoc Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Ken Chen, Greg Corrado, Jeff Dean, and Andrew Ng, "Building high-level features using large scale unsupervised learning," in International Conference on Machine Learning (ICML), 2012, pp. 81–88.

[20] Jeff Dean, Greg Corrado, Rajat Monga, Ken Chen, Matthieu Devin, Quoc Le, Mark Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, and Andrew Ng, "Large scale distributed deep networks," in Advances in Neural Information Processing Systems (NIPS), 2012.

[21] Rich Caruana, "Multitask learning," Machine Learning, vol. 28, pp. 41–75, 1997.

[22] Georg Heigold, Vincent Vanhoucke, Andrew Senior, Patrick Nguyen, Marc'Aurelio Ranazaot, Matthieu Devin, and Jeff Dean, "Multilingual acoustic models using distributed deep neural networks," in Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, Canada, Apr. 2013, IEEE, vol. 1.

[23] Mitchel Weintraub, "Keyword spotting using sri's decipher large vocabulary speech recognition system.," in Proceedings of the International Conference on Acoustics, Speech and Sig- nal Processing (ICASSP). IEEE, 1993, vol. 2, pp. 463–466.