WHOLE SENTENCE NEURAL LANGUAGE MODELS

Yinghui Huang, Abhinav Sethy, Kartik Audhkhasi, Bhuvana Ramabhadran

IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

ABSTRACT

Recurrent neural networks have become increasingly popular for the task of language modeling achieving impressive gains in state-of-the-art speech recognition and natural language processing (NLP) tasks. Recurrent models exploit word dependencies over a much longer context window (as retained by the history states) than what is feasible with n-gram language models. However the training criterion of choice for recurrent language models continues to be the local conditional likelihood of generating the current word given the (possibly long) word context, thus making local decisions at each word. This locally-conditional design fundamentally limits the ability of the model in exploiting whole sentence structures. In this paper, we present our initial results at *whole sentence* neural language models which assign a probability to the entire word sequence. We extend the previous work on whole sentence maximum entropy models to recurrent language models while using Noise Contrastive Estimation (NCE) for training, as these sentence models are fundamentally unnormalizable. We present results on a range of tasks: from sequence identification tasks such as, palindrome detection to large vocabulary automatic speech recognition (LVCSR) and demonstrate the modeling power of this approach.

Index Terms— Whole-sentence language models, Unnormalized models, Recurrent neural network

1. INTRODUCTION

Most statistical language models (LMs) including n-gram LM and standard recurrent neural network LMs are conditional models, that estimate the probability of a word given the previous word sequence. The probability of a sentence s of T words $w_1, w_2, \ldots, w_T$ is calculated as the product of word probabilities by using the chain rule,

$$p(s) = p(w_1, \cdots, w_T) = \prod_{t=1}^{T} p(w_t|h_t) \tag{1}$$

where $h_t = w_1, ..., w_{t-1}$ is the history of word w_t. This history is often truncated to the previous $n-1$ words since long histories are rarely observed in the training data. This includes, for example, n-gram LMs and feed-forward neural network LMs. n-gram LMs estimate the conditional probability of the next word given the history using counts computed from the training data. On the other hand, feed-forward neural network LMs embed the history into a continuous space and use a neural network to estimate this conditional probability. A key drawback of conditional LMs is that the captured context is dependent on the length of the history, which is usually very limited, often less than five words.

1.1. RNN/LSTM language models

Recently, recurrent neural network (RNN) LMs have become popular due to their ability to capture longer word histories than n-gram based LMs [1]. A RNN can be thought of as a feed-forward neural network that is cloned across time with the hidden state at time step $(t-1)$ concatenated with the embedding of the word w_t to form the input that predicts the next word w_{t+1}. Hence, the conditional likelihood of w_{t+1} is influenced by the hidden states at all previous time steps $1, \ldots, t$, thus capturing a very long context. In practice, the history is truncated to the previous 15-20 words in order to speed-up training and decoding.

Often, RNN's use a long short-term memory (LSTM) [2] cell instead of simple sigmoidal hidden neurons. The LSTM is able to capture even longer temporal contexts without suffering from the exploding or vanishing gradient problems prevalent in RNNs. Despite the ability of RNNs to capture much longer temporal context as compared to n-gram LMs or feed-forward neural networks, they still use the chain rule to estimate the probability of the entire sentence. This locally-conditional design fundamentally limits the ability of the model in exploiting whole sentence structures [3]. In this paper we explore whole sentence recurrent language models which are free from the locally-conditional constraints.

1.2. Whole Sentence Language Models

Unlike locally-conditional language models, whole sentence models directly model the probability of a sentence or a utterance. [4] introduced whole sentence maximum entropy language models. As shown in a follow-up paper [5], the model illustrates great capability in integrating any computable property of sentences. It is conceptually simple and straightforward to use in applications.

In this paper, we extend the concept of whole-sentence models to recurrent neural network LMs, starting with a review of related work in section 2. In section 3, we give a detailed description of the neural network architectures that we have explored. Section 4 presents training methods used in this paper. Similar to [5], we apply noise-contrastive estimation (NCE), a sampling-based method to train an unnormalized model. In Section 5, we present results on several tasks, ranging from simple sequence identification tasks to speech recognition. We summarize our findings in section 6.

2. RELATED WORK

Whole sentence maximum entropy models were first proposed in [4] and later improved upon in [5]. These models allowed the flexibility of having custom sentence-level features such as length of sentence which are hard to model via locally-conditional language models [3]. Our work in this paper can be seen as an extension of these models to neural net language models with some modifications such as the use of Noise Contrastive Estimation for unnormalized joint density estimation and not relying heavily on custom sentence-level features.

The idea of sentence-level neural net models has been explored in recent work on sentence classification [6] and on the construction of sentence embeddings [7] [8]. Sentence embedding methods aim to project sentences into a fixed-dimensional vector space where sentences that are semantically-related are close to each other. Recently, [9] proposed general-purpose, paraphrastic sentence embeddings by starting with standard word embeddings and constructing sentence embeddings by training a simple word averaging model.

Sentence classification which assign a sentence to a class label, for example, positive or negative review on movies [10] are also conditional models, whereas our focus in this paper is to assign a probability to each sentence. The proposed whole sentence neural model therefore aims to assign higher scores to sentences that are more likely to occur in the domain of interest.

An alternative task-based approach to whole sentence modeling is Discriminative Language Models (DLM) for ASR [11] [12] and machine translation (MT) [13]. Discriminative language models do not attempt to estimate a generative model over strings. Instead, they are trained on the output of the ASR or the MT system with the ground truth, in an attempt to directly optimize WER or BLEU score respectively. Thus DLMs cannot be trained on just textual training data but required matched speech data and transcriptions for ASR and aligned bilingual data for MT.

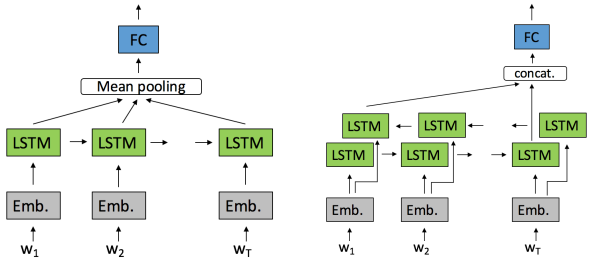
3. WHOLE SENTENCE NEURAL MODELS

In this paper, we focus on training a language model which predicts the probability of a sentence $p(s)$ directly without computing conditional probabilities for each word in the sentence independently. We focus on using recurrent LSTM models in this paper as the neural architecture of choice. It should be noted that the proposed approach is flexible and can be extended to other architectures such as convolutional neural networks easily. We present two simple LSTM architectures that provided good empirical results.

In the first network shown in Figure 1a, the representation vector of a sentence is generated from a one-layer, uni-directional LSTM followed by mean-pooling over the hidden states at each time step. The second network shown in Figure 1b is a one-layer bidirectional LSTM. We stack the hidden representations of the last state in each direction to form the sentence embedding. In both networks, a fully-connected linear layer is added to the LSTM layer to obtain the final score of the sentence. Note that in contrast to typical word LSTM language models,

- Our model does not require any softmax computation to compute conditional probabilities of individual words.
- The model generates a single output score for the whole sentence which we treat as an un-normalized probability.

Given the fundamentally un-normalized nature of the model, we need a sampling-based approach to estimate model parameters. While prior work on whole sentence maximum entropy models [5] used importance sampling to estimate model parameters, we use noise contrastive estimation (NCE) [14], which has been shown to be an effective technique to speed-up training of neural language models [15]. In the next section 4, we will delve into the details of how we use NCE for training the whole sentence neural language model.



(a) LSTM (b) BiLSTM

Fig. 1: Recurrent Neural Network architectures explored in this paper.

4. TRAINING OF WHOLE SENTENCE MODELS

4.1. Noise Contrastive Estimation

Noise contrastive estimation (NCE) was first introduced as a sampling-based approach for unnormalized training [14] of statistical models. Rather than maximizing the likelihood of the training data, a number of noise samples are generated for each training sample. Subsequently, the parameters of the model are trained to maximize the likelihood of a binary prediction task that identifies the ground truth from the noise samples. In other words, NCE performs a nonlinear logistic regression to discriminate between the observed training data (ground truth) and artificially-generated noise data.

Mathematically, let $X = (x_1, x_2, \ldots, x_S)$ be the S sentences in training data. We denote by $Y = (y_1, y_2, \ldots, y_{\nu S})$ the $\nu * S$ samples that are drawn from a noise sampler model with probability density of $p_n(\cdot)$, where $\nu > 1$. We also denote the density estimate of the model by $p_m(\cdot; \theta)$. Then the NCE loss is defined as

$$l(\theta) = \sum_{i=1}^{S} \ln[h(x_i; \theta)] + \sum_{i=1}^{\nu S} \ln[1 - h(y_i; \theta)] \qquad (2)$$

where

$$h[u; \theta] = \frac{1}{1 + \nu \exp\left(-G(u; \theta)\right)}$$

and $G(u; \theta)$ is the log-odds ratio between $p_m(\cdot; \theta)$ and $p_n(\cdot)$, i.e. $G(u; \theta) = \ln p_m(u; \theta) - \ln p_n(u)$. By optimizing the loss function (2) with model parameters θ, it can be shown that the model p_m learns the probability density of X in the limit [14].

NCE has been used extensively for improving the scalability of conditional recurrent neural network based language models by speeding-up the expensive computation of the normalization term (softmax function) in the output layer [15, 16, 17, 18]. During NCE-based training of neural network language models, only the connections associated with a few words in the output layer need to be considered, thereby eliminating the need to compute the normalization over the full output vocabulary. As noted in [18], NCE implicitly constrains the variance of the normalization term to be very small during training, which make it feasible to use these *unnormalized* probabilities during testing. With sufficient number of noise samples the solution to the binary prediction model converges to the maximum likelihood estimate on the training data. A modified NCE algorithm using negative sampling was proposed in [19] to learn accurate representations of frequent words and phrases.

We adapt NCE to whole sentence models by sampling entire sentences from a *noise* model as opposed to word samples used for speeding-up the softmax computation. This makes the sampling process more complex, as we discuss in detail in the next subsection. Another important distinction from the conditional case is that the true data and noise samples do not share the same context representations. This implies that the entire model and not just the final layer output needs to be recomputed for the noise samples.

4.2. Sampling for NCE

In this paper, we use back-off n-gram language models built on the training data as noise samplers. Noise samples were drawn in the following two ways:

- Sampling Type-I: We generated word sequences using the noise sampler model, such as, an n-gram or LSTM language model.
- Sampling Type-II: We first randomly select one sentence from the training data, and then randomly select N positions to introduce an insertion, substitution, or deletion error in that sentence. This process can be viewed as sampling from an edit transducer [20]. The probability of a word to be inserted or substituted with is assigned by the noise sampler model based on the n-gram history at the position being considered. We ensured that each resulting noisy sentence has an edit distance of at most N words from the original sentence. We assigned a score to each sentence from the noise sampler model, where the score is simply the sum of all n-gram scores in the sentence. Note that using the score directly from the noise sampler instead of computing it using a word-to-word transduction [20] model is an approximation, which we found to work well in practice.

Sampling Type-I is theoretically-correct to get the right NCE loss. However, one of the drawbacks of generating sentences from the n-gram model is that n-gram models almost always prefer shorter sentences. Hence, it would be hard to cover the noise space and this would reduce generalization over the kinds of errors typically encountered in speech recognition tasks. Hence, we also experimented with Sampling Type-II for certain types of large vocabulary, spontaneous speech recognition tasks (Section 5.2).

5. EXPERIMENTAL SETUP AND RESULTS

In this section, we describe the various tasks against which the whole sentence language model described in Section 3 is evaluated.

5.1. Sequence Identification tasks

5.1.1. Data sets

The first set of tasks we considered were tasks in which we explored the ability of the model to fit sentences that are based on algorithmic string generation or on a grammar. While these tasks appear to be hand-picked, they help us to validate our idea that the whole sentence model can detect patterns which rely on the entirety of a sentence and may be hard to capture in a locally-conditional model. The tasks that we investigate are:

- Palindrome detection (PAL). We generated a 1M word corpus of palindromes from a 10-word vocabulary . Each sentence in the corpus contains 10 words. Examples include sentences such as, *The cat ran fast ran cat the*, *one five six nine six five one*, etc.
- Lexicographically-ordered words (SORT): We generated a 1M words corpus from a 15-word vocabulary. Each sentence is of length of 10 words and contains words in lexicographically sorted order i.e for a sentence $w_1 ... w_n$, for all words if $i < j$ then $w_i < w_j$ in the lexicographic sense.
- Expressing dates (DATE): We generated a 7M words corpus from a vocabulary of 70 words enumerating various ways to express a date, such as, *January first nineteen oh one*.

The network configuration presented in Fig. 1b is used to train the whole sentence model for these tasks. The BILSTM has an embedding size of 200 with 700 hidden units. The noise samples are generated by a 4-gram model and noise sampling allowing for one edit (substitution) in the noisy sample sentence. The model was trained using stochastic gradient descent and the NCE loss function with a mini-batch size of 512. For every epoch, we generated a set of 20 noise samples per data point. The learning rate was adjusted using the annealing strategy originally described in [21] where the learning rate was halved if the heldout loss was worse than the previous iteration. The model converged after roughly 8 to 10 iterations.

5.1.2. Results

For each of the sequence identification tasks, 10% of the generated data was used as the test set and excluded from training. Imposter sentences were generated by substituting one word in any position in the true samples in the test set. The accuracy of the model is evaluated on its ability to classify the true sentences from their imposters.

The trained whole sentence model assigns a score to each sentence in the test set. A simple linear classifier on these scores assigns the sentence to one of the two classes. The performance of the model is evaluated by measuring its classification accuracy. Surprisingly, for all three tasks, its accuracy on an average is above 99%. This implies that the model is able to differentiate between sequences that fit the algorithmic pattern and the imposters. However, it should be noted that there is an inherent structure to these grammar-like tasks in contrast to free-form text, which makes it easy for the model to learn a structure.

We take a close look at the DATE test set to see how the model performs compared with a general n-gram model. Based on the true samples (REF) in test set, we generate 4 types of noise samples, by substituting (SUB), inserting (INS) or deleting (DEL) one word at any position from the test utterance. The fourth noise sample is generated by the n-gram model (RAND). One set of reference and noise samples is given in Table 1. The classification error rate of the 4-gram model and the sentence model are presented in Table 2. While it is expected that the n-gram model will have a relatively high error rate when the noise samples are generated by itself (RAND), the sentence model achieves significantly better performance. We hypothesize that this is because the sentence model does not make conditional independence assumptions inherent in the locally-conditional models.

Table 1: Example sentences from the DATE test set

REF	July the twentieth nineteen seventy nine
SUB	July twenty twentieth nineteen seventy nine
INS	July the twentieth nineteen ninety seventy nine
DEL	July the twentieth * seventy nine
RAND	July the twenty seventh of September two thousand eighteen

Table 2: Classification error rate (%)

	4-gram	sentence model
REF	0.03	0.00
SUB	0.73	0.04
INS	0.01	0.00
DEL	2.22	0.00
RAND	22.70	0.40

5.2. Large Vocabulary Speech recognition

Next, we evaluate the modeling power of the whole sentence model on two speech recognition tasks. In most speech recognition systems, the decoder can generate alternate hypotheses (N-best lists) from lattices during a first recognition pass. These N-best lists are normally rescored by a complex neural network language model such as an RNN or LSTM to find the best scoring path in the N-best lists. We present results by rescoring 100 N-best lists with the whole sentence model on two speech recognition tasks.

- *Conversational Interaction (CI):* This is a task seen in spoken language systems with utterances comprising of digits and alphabets in insolation, command phrases and short dialogs. The acoustic model was trained on 3600 hours of audio data consisting of clean and noisy utterances from public corpora such as, broadcast news, Mixer 6 [22], the AMI meeting corpus [23] and in-house, conversational interaction data. This corpora is further augmented with realistic environmental noises from the JEIDA corpus [24] and impulse responses from RWCP [25] at various SNRs between 5 to 20 dB to total 3600 hours. The decoder used a vocabulary of 275K and a word 4-gram LM with 150M ngrams (Perplexity:72.12). The baseline one-best word error rate of this system is 8.5%. The test set is of duration 1.5 hours consisting of accented data ($\sim$10 accents) covering spoken interaction in concierge and other similar application domains.
- *Conversational telephony speech transcription (SWB):* This task is the well-benchmarked Hub5-2000 testset from the NIST Hub5 2000 evaluation. The N-best lists for this task were generated by a state-of-the-art system described in [26]. The acoustic model consists of a score fusion of a bidirectional LSTMs and a convolutional residual net (ResNet) models trained on 2000 hours. The decoder used a vocabulary of 85K and a word 4-gram LM with 24M words (perplexity = 101). The test set is 2.5 hours in duration. The baseline one-best word error rate of this system is 6.9%.

The baselines used for both tasks are state-of-the-art and are very strong baselines to improve upon.

Whole sentence model configuration: For the SWB task, the whole sentence LM is trained on only the transcripts of the training data, i.e. 24M words. For the CI task, the model was trained on even less data, approximately 50 hours of conversation totalling 440K words. The network presented in 1a is used with a size of 512 for both projection and hidden layer on SWB task, 265 on CI. Noise samples were drawn from a bigram language model.

Given the spontaneous style of the SWB task, we explored noise samples drawn from 1-edit up to 3-edit to allow for enough diversity. From the results presented in Table 3, it can be seen that the full sentence model can provide gains over a word-LSTM trained on the the same data as the baseline LM [26]. When the N-best lists are rescored with a word-LSTM, the SWB task improves by 0.5, while there is no improvement on the CI task. However, modeling the whole sentence seems to buy modest gains on both tasks, suggesting that model has indeed captured longer context than a simple word LM. We present a few examples in Table 4 from both tasks. It can be seen that the sentence model is able to capture sufficient long-term context and correct few errors that could make a bigger difference to downstream natural language processing applications.

Table 3: Word Error Rate (%) on SWB and Conversational Interaction task

	SWB	CI
n-gram	6.9	8.5
+ word LSTM	6.5	8.5
+ sentence model	6.3	8.3

Table 4: Examples from the SWB and CI tasks illustrating the type of errors recoverable by the proposed sentence LM model (Errors are marked in red)

SWB	
Ref	actually we were looking at the saturn S L two
n-gram	actually we were looking at the saturday I sell to
word LSTM	actually we were looking at the saturday S L too
Sentence LM	actually we were looking at the saturn S L too
CI	
Ref	what are the famous attractions near here
n-gram and word LSTM	what are the famous attraction is near here
Sentence LM	what are the famous attractions near here
Ref	could you send some soda to room three four five
n-gram and word LSTM	could you send some sort of to room three four five
Sentence LM	could you send some soda to room three four five

6. SUMMARY

In this paper, we present our initial results at *whole sentence* neural language models building upon prior work on whole sentence maximum entropy models. These models estimate the probability of the entire word sequence directly without partial computations of individual word likelihoods. To avoid normalizing over the whole sentence space, we apply NCE for training our recurrent models. Results are presented on a range of tasks: from sequence identification tasks such as, palindrome detection to large vocabulary automatic speech recognition (LVCSR) and conversational interaction. We demonstrate the modeling power of this approach. On these preliminary experiments, the modest gains over state-of-the-art baselines suggest that the recurrent neural network has the ability to better model full sentences.

7. ACKNOWLEDGEMENT

We would like to thank our colleagues Stanley Chen, Hong-Kwang J Kuo, George Saon and Gakuto Kurata from the IBM T. J. Watson Research center for their invaluable support.

8. REFERENCES

[1] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur, "Recurrent neural network based language model.," in *Interspeech*, 2010, vol. 2, p. 3.

[2] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[3] Le Hai Son, Alexandre Allauzen, and François Yvon, "Measuring the influence of long range dependencies with neural network language models," in *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, Stroudsburg, PA, USA, 2012, WLM '12, pp. 1–10, Association for Computational Linguistics.

[4] Ronald Rosenfeld, "A whole sentence maximum entropy language model," in *Proceedings of the IEEE Workshop on Speech Recognition and Understanding*, 1997.

[5] Stanley F. Chen and Ronald Rosenfeld, "Efficient sampling and feature selection in whole sentence maximum entropy language models," in *ICASSP, Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, 1999.

[6] Yoon Kim, "Convolutional neural networks for sentence classification," in *EMNLP*. 2014, pp. 1746–1751, ACL.

[7] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler, "Skip-thought vectors," *CoRR*, vol. abs/1506.06726, 2015.

[8] Mingbo Ma, Liang Huang, Bing Xiang, and Bowen Zhou, "Dependency-based convolutional neural networks for sentence embedding," *arXiv preprint arXiv:1507.01839*, 2015.

[9] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu, "Towards universal paraphrastic sentence embeddings," in *Proceedings of ICLR*, 2016.

[10] Bo Pang and Lillian Lee, "A sentimental education: Sentiment analysis using subjectivity," in *Proceedings of ACL*, 2004, pp. 271–278.

[11] Brian Roark, Murat Saraclar, Michael Collins, and Mark Johnson, "Discriminative language modeling with conditional random fields and the perceptron algorithm," in *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004, p. 47.

[12] Erinç Dikici, Murat Semerci, Murat Saraçlar, and Ethem Alpaydin, "Classification and ranking approaches to discriminative language modeling for asr," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 2, pp. 291–300, 2013.

[13] Libin Shen, Anoop Sarkar, and Franz Josef Och, "Discriminative reranking for machine translation.," in *HLT-NAACL*, 2004, pp. 177–184.

[14] Michael U Gutmann and Aapo Hyvärinen, "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 307–361, 2012.

[15] Andriy Mnih and Yee Whye Teh, "A fast and simple algorithm for training neural probabilistic language models," in *Proceedings of the 29th International Conference on Machine Learning*, 2012, pp. 1751–1758.

[16] Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang, "Decoding with large-scale neural language models improves translation.," in *EMNLP*. Citeseer, 2013, pp. 1387–1392.

[17] Abhinav Sethy, Stanley Chen, Ebru Arisoy, and Bhuvana Ramabhadran, "Unnormalized exponential and neural network language models," in *ICASSP*, 2015.

[18] Xie Chen, Xunying Liu, Mark J. F. Gales, and Philip C. Woodland, "Recurrent neural network language model training with noise contrastive estimation for speech recognition," *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5411–5415, 2015.

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., pp. 3111–3119. Curran Associates, Inc., 2013.

[20] Mehryar Mohri, "Edit-distance of weighted automata," in *CIAA*. Springer, 2002, vol. 2, pp. 1–23.

[21] H. Bourlard and N. Morgan, "Generalization and parameter estimation in feedforward nets: Some experiments," in *Advances in Neural Information Processing Systems*, 1990, vol. II, pp. 630–637.

[22] L. Brandchain, "The mixer 6 corpus: Resource for cross-channel and text independent speaker recognition," *LREC*, 2010.

[23] Jean Carletta, "Unleashing the killer corpus: experiences in creating the multi-everything ami meeting corpus," *Language Resources and Evaluation*, vol. 41, no. 1, pp. 181–190, 2007.

[24] S Itahashi, "Recent speech database projects in japan," *Proc. ICSLP*, 1990.

[25] Satoshi Nakamura, Kazuo Hiyane, Futoshi Asano, Takanobu Nishiura, and Takeshi Yamada, "Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition.," in *LREC*, 2000.

[26] George Saon, Gakuto Kurata, Tom Sercu, Kartik Audhkhasi, Samuel Thomas, Dimitrios Dimitriadis, Xiaodong Cui, Bhuvana Ramabhadran, Michael Picheny, Lynn-Li Lim, et al., "English conversational telephone speech recognition by humans and machines," *arXiv preprint arXiv:1703.02136*, 2017.