REDUCING TOTAL LATENCY IN ONLINE REAL-TIME INFERENCE AND DECODING VIA COMBINED CONTEXT WINDOW AND MODEL SMOOTHING LATENCIES

Chandrashekhar Lavania and Jeff Bilmes

Department Of Electrical Engineering
University Of Washington, Seattle-98195, USA

ABSTRACT

Real-time low-latency online inference and decoding in sequential probabilistic models are important in many interactive systems, including automatic speech recognition (ASR) and streaming environments. We study total inference latency (TL) in such systems, the additively combined latency of the inherent look-ahead of a deep neural network's (DNN) contextual window (CWL) in a DNN-HMM hybrid system and the latency incurred during Kalman-style smoothing in a dynamic probabilistic model (MSL) (hence, TL = CWL + MSL). For a fixed TL, the best accuracy can occur with a strictly positive MSL, often by quite a bit, a surprising result given the DNN's power. Furthermore, we find that accuracy is often improved with smaller TL and larger MSL. These results suggest that for optimal low-latency real-time decoding, the size of a DNN context window along with model smoothing should be jointly considered.

***Index Terms*—** Streaming inference, online inference, hybrid models, speech recognition, deep learning

1. INTRODUCTION

Automatic Speech Recognition (ASR) lies at the vanguard of methods for sequential probabilistic inference. Indeed, ASR has been an area of interest for several decades and gained momentum following advancements in the field of deep neural networks (DNNs). Going back to the 1990s [1, 2], ASR has recently undergone significant improvements thanks to the many-layered architectures of DNNs. For instance, Mohamed and Hinton [3] started by using restricted Boltzmann machines for phone recognition, Sivaram and Hermansky [4] used sparse multi-layer perceptrons, and a flood of interesting and novel different architectures of DNNs have recently appeared, such as deep convolutional neural networks [5, 6, 7, 8] and recurrent neural networks [9, 10, 11], to name only a few. As such, deep neural networks are now state-of-the-art, and are regularly used in industry. The methods developed for ASR, moreover, are applicable to any sequential inference problem where one wishes to take a sequence, or in the online-inference case, a stream, of input features and map them to an output sequence.

From an inference perspective, inference can be approached in either an *offline* or an *online* manner. The offline approach assumes that all the signal (the entire utterance in the ASR case, where both future and past are available) is available while performing inference. In contrast to this, the online approach makes no such assumptions, and must make temporally local decisions almost immediately as the data comes in, based only on the present and past with no knowledge of the future. In this work we focus on the latter online approach.

The online mechanism has several important use cases. Online automatic speech recognition has become part of several human-computer interactive scenarios. Such cases need a speech recognition system to recognize the speech utterance with small latency, and therefore offline techniques are not applicable and recognition systems need to perform online inference. Other applications, such as real-time low-latency classification of a stream of sensor signals also falls under this category. It is important to understand that techniques that involve usage of fixed limited amount of future for inference are also online procedures as they do not require the availability of the entire unboundedly long future (i.e., until the end of an utterance). The importance of such online techniques has been recognized in the research community. Kalman smoothing has long been a known technique of probabilistic inference [12] and that is equally applicable to HMMs as to Kalman models. Also, Narsimhan et al [13] discuss several procedures for online decoding in the context of Markov models under latency constraints. Similarly, Bloit and Rodet [14] proposed a short time Viterbi procedure for online inference in HMMs that takes into account a short future. The tradeoff between latency and accuracy is inherent in these contexts, and mechanisms to provide improved accuracy under a fixed amount of latency are useful to study.

Traditionally, ASR systems that utilize hybrid DNN/HMM models involve a neural network that is comprised of a window of past and future observations centered around a current frame t — the context comprises r (radius of the window) frames before and after t, yielding a diameter of $2r+1$. Thus, the window at frame t consists of frames $t-r : t+r$ (i.e., from frame $t-r$ to frame $t+r$, inclusive). Inference latency (defined as the time between when a new observation arrives at time t and when we can make an inference about the category of the feature at time t) thus comes both from window radius (r) at the input of a neural network and also any delay (τ) due to additional future needed for Kalman-style inference, say in an HMM. We call the sum ℓ the *total latency* (TL). Hence, TL = CWL + MSL (equivalently, $\ell = \tau + r$) — the contextual window latency (CWL) is equivalent to r, and the model smoothing latency (MSL) is denoted by τ. We use either the alphabetic symbols (TL, CWL, and MSL) or the mathematical symbols (ℓ, r, τ) appropriate for the current context.

Our Contributions: We empirically demonstrate that for a fixed total latency (TL), a combination of windowed input to a neural network in concert with non-zero model smoothing latency (MSL > 0, or $\tau > 0$ in Equation 2) can produce better real-time phone recognition as compared to zero model smoothing latency setup (MSL $= 0$, equivalent to Equation 1) with only windowed neural network input ($r > 0$). We also demonstrate that setups with smaller TL and larger MSL and can often perform better than setups with larger TL and smaller MSL. We use a standard DNN-HMM style hybrid model and demonstrate this behavior on the TIMIT data set. It is important to understand that this works explores tradeoffs in one specific scenario for speech recognition on TIMIT, which is a scenario similar to many streaming sensor processing tasks, where there are not many (e.g., 10s) labels as in large vocabulary speech recognition (hundreds of thousands). Our primary goal in this work, rather, is to show that

tradeoffs can exist between CWL and MSL.

Outline: We describe the inference methodology in Section 2 and the data used in the experiments in Section 3. The hybrid model for phone recognition is discussed in Section 4. We discuss the empirical results in Section 5 and conclude in Section 6.

2. INFERENCE METHODOLOGY

Traditionally, inference has been approached in either an offline or an online manner. In the offline context that involves knowledge of the whole segment (i.e., speech utterance in ASR), inference in an HMM or graphical model is achieved via Viterbi decoding. In the online Kalman-filter style context, there are two well-defined inference methods $\forall t$, viz. 1) filtering and 2) smoothing:

$$\text{Kalman-Style Filtering:} \quad y_t^* \in \text{argmax}_{y_t} \Pr(y_t | \bar{x}_{1:t}), \tag{1}$$

$$\text{Kalman-Style Smoothing:} \quad y_t^* \in \text{argmax}_{y_t} \Pr(y_t | \bar{x}_{1:t+\tau}), \tag{2}$$

where y_t is the random variable that is being inferred at frame t, $\bar{x}_{1:t}$ are all the observation up to frame t, and y_t^* is the inferred value of y_t . The term τ corresponds to the model smoothing latency (MSL) and can also be called the "smoothing parameter." Equation 1 implies that filtering infers y_t, based upon all observations up to only frame t. Therefore, no information from the future (apart from any information due to context from any windowed observation, as those commonly used in hybrid DNN models) is used in inference. In the Forward-Backward algorithm paradigm, there is no backwards pass from instances of variables in the future. In contrast to this method, smoothing allows a partial backwards pass from variables that are up to τ frames into the future. Furthermore, unlike Viterbi decoding, filtering and smoothing do not, in general, attempt to infer the best sequence of a random variables[1] — i.e., they infer y_t and not $y_{1:T}$ (T being the last frame in a segment). Therefore an inferred value y_t^* via filtering/smoothing need not necessarily correspond to the best sequence $y_{1:T}^*$ at frame t, even at large τ.

When the input consists of a window of observations (as is the case with DNN-HMM models), then $\bar{x}_t = x_{t-r:t+r}$ where x_t is the vector of features at time t (i.e., MFCCs in speech, or some sort of raw feature vector) — hence, $\bar{x}_t$ is really a $m \times (2r+1)$-dimensional matrix of features corresponding to a time radius r window of features and centered at time t, where m is the number of features per time frame (e.g., $m = 39$ is typical in ASR for MFCCs and their delta and double deltas). In such a case, "smoothing" will require knowledge of features of up to $\ell = r + \tau$ actual frames into the future. Figure 1, describes an example with the different latencies encountered while performing online inference in the case where $\tau = 3$ and $r = 1$.

As mentioned above, given the power of DNNs, one might expect that optimal accuracy for any given and fixed $\ell = r + \tau$ is achieved when $\tau = 0$ and $\ell = r$. Our results below show, surprisingly, that optimal accuracy is almost always achieved with $\tau > 0$ for a given fixed ℓ. This implies one should consider both Kalman-style smoothing in addition to a DNN context window for optimal accuracy.

3. THE DATA

The TIMIT database [15] is used for all the experiments in this work. While TIMIT is by no means a state-of-the-art data set in ASR circles, it is still a useful data set from a machine learning perspective [3, 16], and is still widely used. TIMIT, moreover, is also a good surrogate for other forms of time signals (e.g., sensor streams for internet-of-things or human activity recognition applications). In accordance with Lee and Hon [17], the set of 61 TIMIT phone labels are collapsed to a smaller set of 48 phones for training. In addition to those 48 phones, we also include (for training only) one more phone. We retain the glottal stop 'q' as a training label, but ignore it during evaluation (it is removed entirely by Lee and Hon for training and testing). In addition to this, before scoring, the 48 phones are mapped to a set of 39 phones. We also ignore "sil" during scoring. Since the experiments are based upon the phone recognition accuracy, the scoring is done using the methodology of HTK's HResult tool. A phone transition language model is used in all experiments.

The input features used by the hybrid model are either Mel-frequency cepstrum coefficients(MFCCs) or filter bank (FBANK) features with energy over 25.6 ms windows, plus the first-order and second-order temporal differences, giving 39 total features per 10 ms frame for MFCC case and 123 total features for FBANK case.

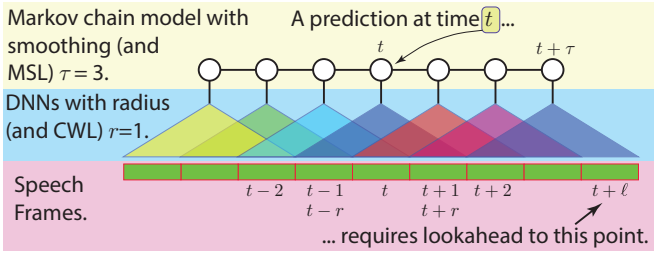


Fig. 1: The different latencies in online inference. Context window latency (CWL) is represented by $r = 1$, model smoothing latency (MSL) is represented by $\tau = 3$, and total latency (TL) is represented by $\ell = r+\tau = 4$. A prediction at time t therefore needs an additional ℓ frames of future look-ahead

4. THE HYBRID MODEL

We build a hybrid DNN-HMM style framework for online phone recognition, and one that can control TL via CWL and MSL. A useful survey of hybrid models is given in Trentin and Gori [18]. This recognition framework is first trained on the TIMIT training set, and the TIMIT development set is used to gauge the phone recognition accuracy for different settings of the CWL (r) and the MSL (τ).

As is standard, the DNN is trained using forced alignments of the phone states, where each phone is assumed to have 3 states. The DNN feeds into a graphical model using the concept of Pearl's virtual evidence [19, 20] and encodes the uncertainty with the data through deep unaries instead of typical Gaussian mixtures — this virtual evidence approach is mathematically equivalent to standard hybrid systems, as the DNN outputs at time t are multiplicatively applied to the Markov state at time t, where a state value has its score multiplicatively modified by the corresponding DNN output probability. As is common practice, the input to the neural network is a length-$2r + 1$ window of contiguous frames. In addition to this, we do not divide the output of the neural network with prior probability over states (an act that would yield scaled likelihoods [1]), as empirically we found this omission resulted in better performance. Our DNNs were trained using the rectified linear (RelU) non-linearity, $f_{rectlin}(z) = \max(0, z)$. The networks also used 20% dropout [21] and were trained using the ADAGRAD [22] procedure.

All our DNNs were trained using Caffe [23], and online inference at different latencies was performed using GMTK [24, 25], which

[1] There are variants that $\forall t$ do repeatedly infer the best sequence backwards from $t + \tau$ back to t but we do not address them here.

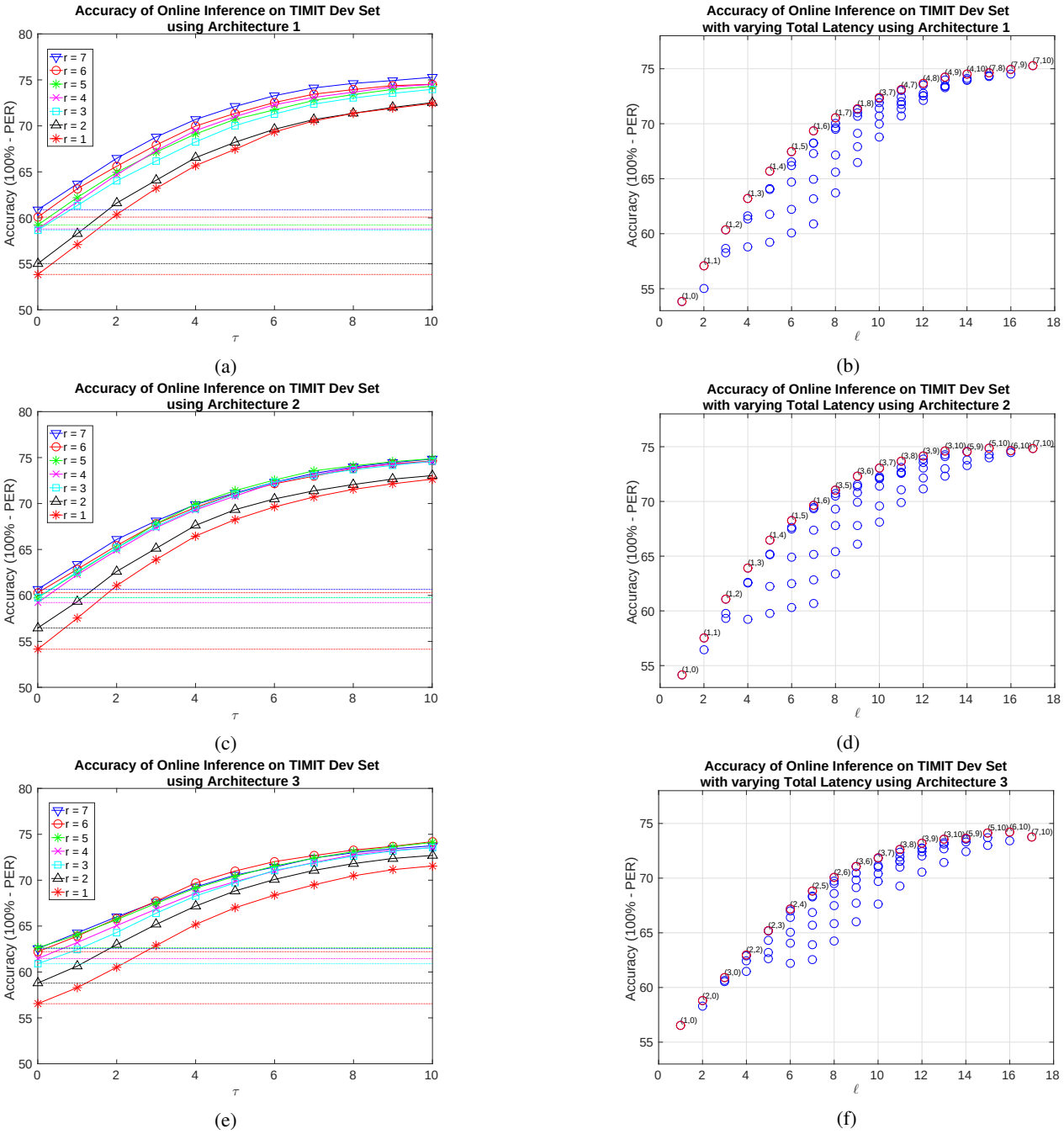
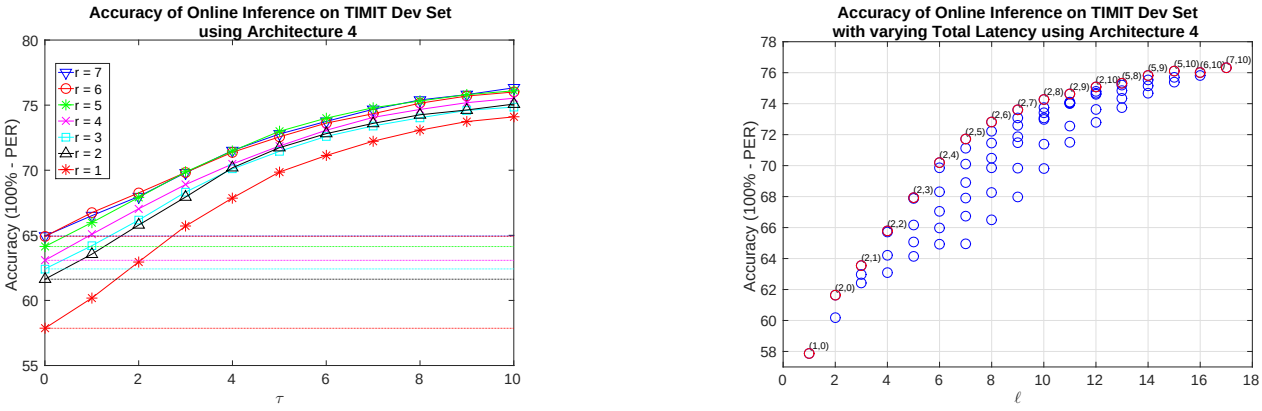


Fig. 2: Performance of fully connected DNN based models. Accuracy over TIMIT development set for different values of model smoothing latency (τ), and various choices of input context window radius (r) of the neural network are depicted for (a) Architecture 1 (10M parameters and 9 hidden layers), (c) Architecture 2 (20M parameters and 9 hidden layers), and (e) Architecture 3 (10M parameters and 5 hidden layers). The dotted horizontal lines denote the accuracy using $\tau = 0$ for various choices of r. Similarly, accuracy over the TIMIT development set with different values of total latency (ℓ) are shown for (b) Architecture 1 (10M parameters and 9 hidden layers), (d) Architecture 2 (20M parameters and 9 hidden layers), and (f) Architecture 3 (10M parameters and 5 hidden layers). The model with highest accuracy for each value of ℓ is marked in red along with the corresponding (r,τ) values, where we clearly see that for the vast majority of cases, it is better to have $\tau > 0$ — in many cases, moreover, τ is a significant component of ℓ!

supports hybrid systems via the mechanism of virtual evidence.[2]

[2]Recent versions of GMTK allow the expression of a DNN-based virtual evidence conditional probability table, where the probability scores of any DNN model can be multiplicatively applied to the scores of corresponding



(a) (b)

Fig. 3: Performance using a convolutional neural network architecture with 1 convolutional layer (filter width = $2r$), 1 pooling layer, and 3 fully connected hidden layers containing 2000 nodes each (Architecture 4). (a) Accuracy over TIMIT development set with different model smoothing latency (τ) for various choices of input window radius (r) of the neural network. The dotted horizontal lines denote the accuracy using $\tau = 0$ for various choices of r. (b) Accuracy over TIMIT development set with different values for total latency (ℓ). The model with highest accuracy for each value of ℓ is marked in red along with the corresponding (r,τ) values

5. RESULTS AND DISCUSSION

In our experiments, we address the following questions: 1) what is the influence of the model smoothing latency (MSL) on models trained with different context window radii (CWL); and 2) is it possible to compensate for smaller input window radii by doing smoothing (Equation 2) instead of filtering (Equation 1).

We explore the performance (accuracy of phone recognition) for radii (CWL) settings varying from 1 to 7 (i.e., a window diameter of 3 to 15 frames). We first use fully connected DNNs with 10M and 20M parameters. A third variation we explore is the DNN depth, where we test 5 and 9 hidden layer models. Overall we denote models with 10M parameters and 9 hidden layers as "Architecture 1", models with 20M parameters and 9 hidden layers as "Architecture 2", and models with 10M parameters and 5 hidden layers as "Architecture 3." Figures 2a, 2c and 2e depict the accuracy of online inference for different variations of the model smoothing latency (τ) using hybrid models incorporating different input window radii (r) and sizes of the deep neural network. We see that the accuracy moves towards saturation with increase in model smoothing latency (τ) and any substantial gains appear for smaller values of τ. It can also be observed that networks with larger r produce similar performance, and again significant improvements are achieved for smaller window radii. This similar behavior of both r and τ intuitively implies the diminishing influence of future events in such time signals.

An even more useful observation stems from the usefulness of model smoothing latency in counteracting the effect of smaller r. In accordance with Equations 1 and 2, it can be argued that filtering is equivalent to smoothing with model smoothing latency set to zero ($\tau = 0$). Therefore Figures 2a, 2c and 2e depict the accuracy of both filtering and smoothing. It can be observed that often a model that uses an input window of smaller radius with smoothing can outperform a model that employs a larger radius window with filtering. For instance, Figure 2a demonstrates that for Architecture 1, a model with $r = 1$, and $\tau = 4$ can outperform a model with $r = 7$ and $\tau = 0$. Therefore, a model with smaller total latency ($r + \tau = 1 + 4 = 5$) outperforms a model with larger total latency ($r + \tau = 7 + 0 = 7$). However, such behavior is not limited to filtering using models with larger r. For example, a model with $r = 3$, and $\tau = 2$ can outperform a model with $r = 6$, and $\tau = 1$ (Figure 2c, Architecture 2), i.e., a model with smaller total latency ℓ (i.e., $\ell = r + \tau$) and larger τ can perform better than a model with larger ℓ and smaller τ. The performances of different settings can also be observed from the perspective of total latency as demonstrated by Figures 2b, 2d and 2f. These figures demarcate the accuracy achieved by the different settings for a given ℓ. It can be observed that the best setting for each ℓ has the radius at most equal to the context ($r \leq \ell$), and often the best accuracy is at $r \ll \ell$. For example, in Figure 2f for $\ell = 7$, the best performance is achieved by the model with $r = 2$ and $\tau = 5$ (red circle). Overall, we see from Figure 2 that our arguments hold for different depths and sizes of fully connected DNNs.

In addition to fully connected DNNs, we also explored convolutional DNNs using the 123 dimensional FBANK features. We denote convolutional neural networks as CNNs, and with 3 layers of hidden nodes as "Architecture 4." Figures 3a and 3b demonstrate that our argument hold trues in this setting as well. These observations imply that for online inference, it can be beneficial to use a combination of model smoothing latency (τ) with smaller input window (r) for faster performance (smaller ℓ).

6. CONCLUSIONS

We show that model smoothing latency (MSL) in conjunction with DNN contextual window latency (CWL) for real-time inference are best considered in tandem to achieve smallest total latency (TL). We also show that a smaller input window to a DNN, along with non-zero lag in Kalman-style smoothing, for online inference, can perform better that a larger DNN window and zero MSL at the same total latency. Our results show that in any sequential decision making context that uses DNNs, it would be imprudent to assume that a DNN's context window is always the only lookahead needed — rather, one should investigate both DNN and Kalman-style lookahead.

states in a dynamic graphical model (DGM). GMTK also now implements offline and online inference, and Kalman-style smoothing for any τ.

7. REFERENCES

[1] Herve A Bourlard and Nelson Morgan, *Connectionist speech recognition: a hybrid approach*, vol. 247, Springer Science & Business Media, 1994.

[2] Yoshua Bengio, Renato De Mori, Giovanni Flammia, and Ralf Kompe, "Neural network-gaussian mixture hybrid for speech recognition or density estimation," in *Advances in Neural Information Processing Systems*, 1992, pp. 175–182.

[3] Abdel-rahman Mohamed and Geoffrey E Hinton, "Phone recognition using restricted boltzmann machines.," in *ICASSP*, 2010, pp. 4354–4357.

[4] Garimella SVS Sivaram and Hynek Hermansky, "Sparse multilayer perceptron for phoneme recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 23–29, 2012.

[5] László Tóth, "Phone recognition with hierarchical convolutional deep maxout networks," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2015, no. 1, pp. 1–13, 2015.

[6] László Tóth, "Combining time-and frequency-domain convolution in convolutional neural network-based phone recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 190–194.

[7] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, and Gerald Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4277–4280.

[8] Tara N Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran, "Deep convolutional neural networks for lvcsr," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 8614–8618.

[9] Alan Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6645–6649.

[10] Haşim Sak, Andrew Senior, and Françoise Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.

[11] Alex Graves and Navdeep Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1764–1772.

[12] Simon S Haykin, Simon S Haykin, and Simon S Haykin, *Kalman filtering and neural networks*, Wiley Online Library, 2001.

[13] Mukund Narasimhan, Paul Viola, and Michael Shilman, "Online decoding of markov models under latency constraints," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 657–664.

[14] Julien Bloit and Xavier Rodet, "Short-time viterbi for online hmm decoding: Evaluation on a real-time phone recognition task," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 2121–2124.

[15] "TIMIT Acoustic-Phonetic Continuous Speech Corpus Linguistic Data Consortium, Philadelphia," [Online], Available at http://catalog.ldc.upenn.edu/LDC93S1 [Accessed: March 11, 2014].

[16] Jimmy Ba and Rich Caruana, "Do deep nets really need to be deep?," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., pp. 2654–2662. Curran Associates, Inc., 2014.

[17] K. Lee and H. Hon, "Speaker-independent phone recognition using hidden markov models," *Computer Science Department. Paper 1769*, 1988.

[18] Edmondo Trentin and Marco Gori, "A survey of hybrid ANN/HMM models for automatic speech recognition," *Neurocomputing*, vol. 37, no. 14, pp. 91–126, 2001.

[19] Judea Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, Morgan Kaufmann, 1988.

[20] Jeff Bilmes, "On soft evidence in bayesian networks," Tech. Rep. UWEETR-2004-0016, University of Washington, Dept. of Electrical Engineering, 2004, https://www.ee.washington.edu/techsite/papers/refer/UWEETR-2004-0016.html.

[21] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[22] John Duchi, Elad Hazan, and Yoram Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[23] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2014, pp. 675–678.

[24] Jeff Bilmes and Geoffrey Zweig, "The graphical models toolkit: An open source software system for speech and time-series processing," in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, 2002, vol. 4, pp. IV–3916.

[25] Jeff Bilmes, "GMTK: The graphical models toolkit documentation," 2015, https://melodi.ee.washington.edu/gmtk/.