Neural Network Based Pitch Tracking in Very Noisy Speech

Kun Han, *Student Member, IEEE*, and DeLiang Wang, *Fellow, IEEE*

***Abstract*—Pitch determination is a fundamental problem in speech processing, which has been studied for decades. However, it is challenging to determinate pitch in strong noise because the harmonic structure is corrupted. In this paper, we estimate pitch using supervised learning, where the probabilistic pitch states are directly learned from noisy speech data. We investigate two alternative neural networks modeling pitch state distribution given observations. The first one is a feedforward deep neural network (DNN), which is trained on static frame-level acoustic features. The second one is a recurrent deep neural network (RNN) which is trained on sequential frame-level features and capable of learning temporal dynamics. Both DNNs and RNNs produce accurate probabilistic outputs of pitch states, which are then connected into pitch contours by Viterbi decoding. Our systematic evaluation shows that the proposed pitch tracking algorithms are robust to different noise conditions and can even be applied to reverberant speech. The proposed approach also significantly outperforms other state-of-the-art pitch tracking algorithms.**

***Index Terms*—Deep neural networks (DNNs), pitch estimation, recurrent neural networks (RNNs), supervised learning, viterbi decoding.**

I. Introduction

PITCH, or fundamental frequency ($F0$), is one of the most important characteristics of speech signals. A pitch tracking algorithm robust to background interference is critical to many applications, including speaker identification [1] and speech separation [14]. Although pitch tracking has been studied for decades, it is still challenging to estimate pitch from speech in the presence of strong noise, where the harmonic structure of speech is severely corrupted.

A typical pitch determination algorithm consists of two stages. The first stage determines pitch candidates or computes the pitch probability for each time frequency unit. To deal with noise, previous studies either utilize signal processing to attenuate noise [6], [11] or employ statistical methods to model the harmonic structure [42], [5], [22]. However, the selection of pitch candidates is often ad hoc, and it may be less optimal to make a hard decision for pitch candidate selection. Statistical modeling usually relies on strong assumptions, which make the algorithms difficult to generalize to complex acoustic environments. In the second stage, the pitch candidates or probabilities are connected into pitch contours using dynamic programming [5], [11] or hidden Markov models (HMMs) [25], [42].

It is sensible to formulate the pitch determination problem as an HMM decoding problem, where a hidden state corresponds to a pitch frequency and an observation corresponds to acoustic features. This way, pitch determination is equivalent to finding the optimal sequence of hidden states given an observation sequence. In an HMM, a key problem is to estimate the posterior probability given the observation in each time step. In this study, we propose to supervisedly learn the posterior probability that a frequency bin is pitched given the observation.

A deep neural network (DNN) is a feed-forward neural network with more than one hidden layer [17], which has been successfully used in signal processing applications [32], [40]. In automatic speech recognition, the posterior probability of each phoneme state is modeled by a DNN. We adopt this idea for pitch tracking, i.e., we use a DNN to model the posterior probability of each pitch state given the observation in each frame. The DNN is expected to generate accurate probabilistic outputs due to its powerful learning capacity.

Further, speech has prominent temporal dependency which provides rich information for speech processing. A straightforward method to capture temporal information is to include neighboring frames into an expanded feature vector. However, this technique can only capture the temporal information within a limited span, because the dimensionality of the feature is proportional to the number of the frames and it is difficult to train a model with very high dimensional features. To utilize temporal dynamics, a more systematic approach is to directly encode temporal information into learning machines. A recurrent neural network (RNN) is an extension of the feedforward neural network, where the hidden units have delayed self-connections. These recurrent connections allow the network to encode temporal information suitable for modeling nonlinear dynamics. Recent studies have shown promising results using RNNs to model sequential data [30], [39]. Given that speech is inherently a sequential signal and temporal dynamics is crucial to pitch tracking, we consider RNNs to model the probability distribution of pitch states.

To recapitulate, we investigate DNN and RNN based supervised methods for pitch tracking in very noisy speech. With

Manuscript received February 17, 2014; revised June 01, 2014; accepted October 01, 2014. Date of publication October 14, 2014; date of current version October 23, 2014. This work was supported in part by the Air Force Office of Scientific Research (AFOSR) under Grant FA9550-12-1-0130 and the Ohio Supercomputer Center. A preliminary version of this work was presented at ICASSP 2014 [15]. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Zhen-Hua Ling.

K. Han is with the Department of Computer Science and Engineering, The Ohio State University, Columbus, OH 43210 USA (e-mail: hank@cse.ohio-state.edu).

D. L. Wang is with the Department of Computer Science and Engineering and the Center for Cognitive and Brain Sciences, The Ohio State University, Columbus, OH 43210 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASLP.2014.2363410

2329-9290 © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

proper training, both DNN and RNN are expected to produce reasonably accurate probabilistic outputs for pitch states. With the pitch state probability in each frame, a Viterbi decoding algorithm will be utilized to form continuous pitch contours (see also [42]).

This paper is organized as follows. The next section relates our work to previous studies. Section III discusses the feature extraction part. The details of the proposed pitch tracking approach are presented in Section IV. The experimental results and comparisons are presented in Section V. We discuss related issues and conclude the paper in Section VI.

II. Related Prior Work

Recent studies on robust pitch tracking have explored either harmonic structure in the frequency domain, periodicity in the time domain, or the periodicity of individual frequency subbands in the time-frequency domain.

In the frequency domain, harmonic structure exhibits rich information about pitch. Previous studies extract pitch from the spectrum of speech, by assuming that each peak in the spectrum corresponding to a potential harmonic [35], [16]. SAFE [5] utilizes prominent signal-to-noise ratio (SNR) peaks in speech spectra to model the distribution of pitch using a probabilistic framework. PEFAC [11] combines nonlinear amplitude compression to attenuate narrowband noise and chooses pitch candidates from the filtered spectrum.

Another type of approaches utilizes the periodicity of speech in the time domain. RAPT [37] calculates the normalized autocorrelation function (ACF) and chooses the peaks as the pitch candidates. The YIN [6] algorithm uses the squared difference function based on ACF to identify pitch candidates.

An extension of time-domain approaches extracts pitch using the periodicity of individual subbands in the time-frequency domain. Wu *et al.* [42] model pitch period statistics on top of a channel selection mechanism and use an HMM for extracting continuous pitch contours. Jin and Wang [25] use cross-channel correlation to select reliable channels and derive pitch scores from resulting summary correlogram. Huang and Lee [22] compute a temporally accumulated peak spectrum to estimate pitch. Lee and Ellis [28] extract the ACF features and train a multilayer perceptron (MLP) classifier on the principal components of the ACF features for pitch detection.

Different from the above methods, we use spectral domain features to provide a robust representation for pitch tracking in noise. Further, our approach utilizes advanced classifiers, namely deep neural networks and recurrent neural networks, which generate accurate probabilistic pitch states and boost the pitch tracking performance. In addition, we believe that a large dataset with multiple conditions benefits robustness of the proposed algorithms to noises and reverberation.

III. Feature Extraction

The proposed pitch tracking algorithms first extract spectral domain features in each frame, and then employ neural networks to compute the posterior probability of the pitch state for each frequency bin. With probabilistic outputs, we use Viterbi decoding to connect pitch states and form final pitch contours.

The features used in this study are extracted from the spectral domain based on [11]. We compute the log-frequency power spectrogram and then normalize to the long-term speech spectrum to attenuate noises. A filter is then used to enhance the harmonicity.

Specifically, a signal is first decomposed to the spectral domain using short time Fourier transformation. Let $X_t(f)$ denote the power spectral density (PSD) of the frame t in the frequency bin f. The PSD in the log-frequency domain can be represented as $X_t(q)$, where $q = \log f$. Then, the normalized PSD can be computed as:

$$X_t'(q) = X_t(q)\frac{L(q)}{\overline{X}_t(q)} \tag{1}$$

where $L(q)$ represents the long-term average speech spectrum, and $\overline{X}_t(q)$ denotes the smoothed averaged spectrum of speech, which is calculated by using a 21-point moving average filter in the log-frequency domain and averaging over the entire sentence (2 $\sim$ 4 s duration) in the time domain in this study. With the normalized spectrum, we further enhance harmonicity for pitch tracking using a filter with broadened peaks having an impulse response defined as:

$$h(q) = \begin{cases} \frac{1}{\gamma - \cos(2\pi e^q)} - \beta, & \text{if } \log(0.5) < q < \log(K + 0.5) \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where β is chosen so that $\int h(q)dq = 0$, and γ controls the peak width which is set to 1.8.

The convolution $\tilde{X}_t(q) = X_t'(q) \star h(q)$ contains peaks corresponding to harmonics and their multiples and submultiples. Only the spectral components in the plausible pitch frequency range (60 to 400 Hz in this study) are selected as features. So we have a spectral feature vector in frame t:

$$\tilde{\mathbf{x}}_t = (\tilde{X}_t(q_1), \ldots, \tilde{X}_t(q_n))^T$$

Gonzalez and Brookes [11] proposed to extract the spectral feature $\tilde{\mathbf{x}}_t$ for pitch tracking in noise. Ideally, the pitch, $F0$, can be found by taking the highest peak in $\tilde{\mathbf{x}}_t$. In [11], several highest peaks are chosen for each frame as pitch candidates, and a dynamic programming algorithm is then used to form pitch contours. Although the feature vector is designed to deal with noisy speech, rule-based pitch candidate selection may lose useful information because it simply ignores non-peak spectral information. In our study, we treat $\tilde{\mathbf{x}}_t$ as the extracted feature and employ supervised learning to estimate pitch probability, i.e. to learn the mapping from the features to the pitch frequencies. We expect supervised learning to yield better results.

Since neighboring frames contain useful information for pitch tracking, we incorporate the neighboring frames into the feature vector. Therefore, the final frame-level feature vector is

$$\mathbf{x}_t = (\tilde{\mathbf{x}}_{t-d}, \ldots, \tilde{\mathbf{x}}_{t+d})^T$$

where d is set to 2 in our study.

IV. Learning Pitch State Distribution

Instead of selecting pitch candidates, we employ supervised training approach to learn the posterior probability distribution given the features in each frame. Neural networks have recently achieved large progress in speech processing, and we propose to use two kinds of neural networks to model the probability distribution.

A. DNN Based Pitch State Estimation

Our first method is to use a feedforward DNN. To simplify the computation, we quantize the plausible pitch frequency range into M frequency bins, corresponding to M pitch states $s^1, \ldots, s^M$. We use 24 bins per octave in a logarithmic scale to quantize the plausible pitch frequency range (60 to 400 Hz) into 67 bins, i.e., the quantized frequency of the m th bin is $60 \times 2^{(m-1)/24}$ Hz. In addition, we incorporate a nonpitch state s^0 corresponding to an unvoiced speech or speech-free state. So there are totally 68 states [28].

To train the DNN, each training sample is the feature vector $\mathbf{x}_t$ in the time frame t (and its neighboring frames), and the target is an $(M+1)$-dimensional vector of the pitch states $\mathbf{s}_t$, whose element s_t^i is 1 if the groundtruth pitch falls into the corresponding frequency bin, and 0 otherwise.

In order to learn the probabilistic output, we use cross-entropy as the objective function.

$$\mathcal{L}(\mathbf{y}, \mathbf{x}; \Theta) = -\sum_{m=0}^{M} y_m \ln f_m(\mathbf{x}) \tag{3}$$

where $\mathbf{y} = (y_0, \ldots, y_M)^T$ is the desired output and $f_m(\cdot)$ is the actual output of the mth neuron in the output layer. Θ denotes the parameters we need to learn. The activation function in the hidden layers is the sigmoid function and the output layer uses the softmax function for probabilistic outputs.

The DNN in this study includes three hidden layers with 1600 sigmoid units in each layer, and a softmax output layer whose size is set to the number of the pitch states, i.e., 68 output units. The number of hidden layers and the hidden units are chosen from cross validation (see also Section V-B). We use backpropagation with mini-batch stochastic gradient descent to train the DNN model, and the actual cost in each mini-batch is computed from the summation over multiple training samples using Eq. (3).

The trained DNN produces the posterior probability of each pitch state i: $P(s_t^i|\mathbf{x}_t)$.

B. RNN Based Pitch State Estimation

The DNN based method utilizes frame-level features to compute the posterior probabilities of pitch states. Although it utilizes neighboring frames to incorporate temporal information, it is not able to capture long-term temporal dynamics due to the limit of feature dimensionality. As temporal continuity and variation are important characteristics of pitch, we explore a more intrinsic method to capture temporal context information.

An RNN is a natural extension of a feedforward network. In an RNN, the depth comes from not only multiple hidden layers but also unfolding layers through time. An RNN is capable of capturing the long-term dependencies through connections between hidden layers. These attributes have inspired us to use RNNs to model pitch dynamics. One of the key challenges for using RNNs is that training with long-term dependencies can be quite difficult and some new approaches have been proposed to address the problem [36]. In our study, we use a classic RNN [8] and learn the model with truncated backpropagation through time (BPTT) [34], [41].

The RNN has hidden units with delayed connections to themselves, and the output $\mathbf{y}_o = (y_1, \ldots, y_n)^T$ of the RNN at the time step t can be represented as:

$$\begin{aligned}
\mathbf{y}_o(t) &= \psi(\mathbf{W}_{o,j}^T \mathbf{h}_j(t)) \\
\mathbf{h}_j(t) &= \phi(\mathbf{v}_j(t)) \\
\mathbf{v}_j(t) &= \mathbf{W}_{j,j-1}^T \mathbf{h}_{j-1}(t) + \mathbf{W}_{j,j}^T \mathbf{h}_j(t-1) \\
\mathbf{h}_1(t) &= \phi(\mathbf{W}_{1,i}^T \mathbf{x}_i(t))
\end{aligned} \tag{4}$$

where ϕ and ψ are the sigmoid function and the softmax function respectively. $\mathbf{W}_{j,j-1}$ denotes the weights matrix from the $j-1$th hidden layer to the jth hidden layer, and the numbers of the rows and the columns are equal to the number of the units in the $j-1$th layer and the jth layer, respectively. $\mathbf{h}_j$ is a column vector corresponding to the activations of the jth hidden layer. $\mathbf{W}_{j,j}$ denotes the self-connections in the jth layer. Note that, since each unit only has a recurrent connection to itself, $\mathbf{W}_{j,j}$ is a diagonal matrix. For a non-recurrent hidden layer, $\mathbf{W}_{j,j} = \mathbf{0}$. $\mathbf{W}_{o,j}$ specifies the weight matrix between the last hidden layer and the output layer, and $\mathbf{W}_{1,i}$ the weight matrix between the input layer and the first hidden layer. For a recurrent hidden layer, the state of a neuron is influenced by not only the external input to the network but also the network activation from the previous time steps.

With recursion over time on hidden units, an RNN can be unfolded through time and can be viewed as a very deep network with T layers, where T is the number of time steps. The structure of the RNN in our study includes two hidden layers. Each hidden layer has 256 hidden units and only the units in the second hidden layer have self-connections. The input and the output layers are the same as in the DNN.

To use the truncated BPTT to train the RNN, each training sentence is truncated into multiple segments with a fixed length of T frames. Each segment is treated as a sequential training sample and fed into the neural network. To train the network, the RNN is unfolded for T time steps, and the backpropagated error $\delta_j(t)$ for a neuron in the recurrent layer j is computed from both the next layer $\delta_{j+1}(t)$ and the next time step $\delta_j(t+1)$. Although the truncated BPTT cannot capture the temporal information exceeding T time steps, the training is relatively easy. In our experiment, we set $T = 15$ and a longer T does not significantly improve the performance.

In the test phase, the output of the RNN is computed sequentially, and the output of the RNN in the tth frame is the posterior probability $P(s_t^i|\mathbf{x}_1, \ldots, \mathbf{x}_t)$, where the observation is a sequence from the past to the current frame instead of the feature $\mathbf{x}_t$ in the current frame.

C. Viterbi Decoding

The DNN or the RNN produces the posterior probability distribution in each time frame. We then use Viterbi decoding [9],

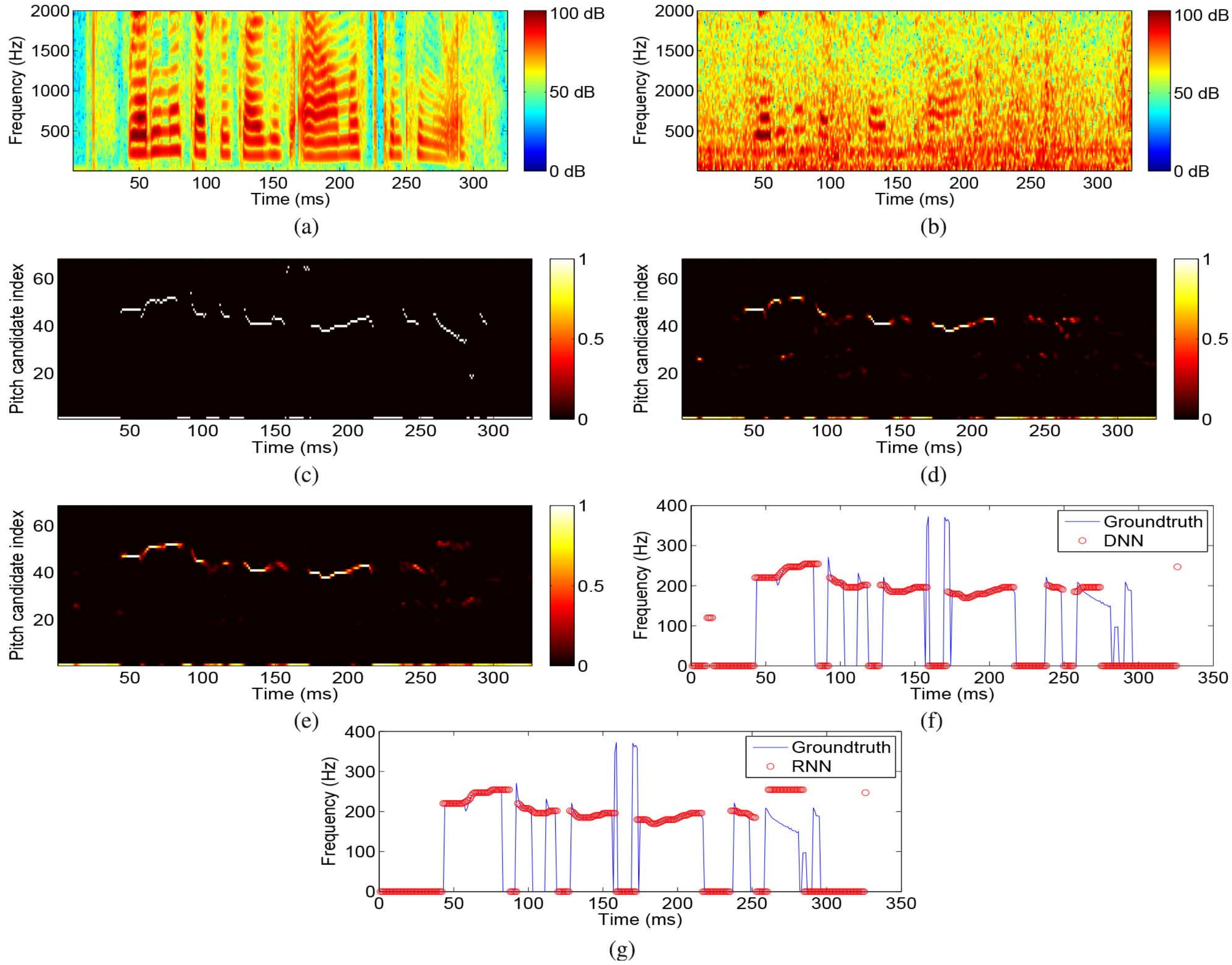

Fig. 1. (Color online) Neural network based pitch tracking. Noisy speech is a female utterance from the TIMIT corpus “Readiness exercises are almost continuous”, mixed with factory noise in -5 dB SNR. (a) Spectrogram of clean speech from 0 to 2000 Hz. (b) Spectrogram of noisy speech from 0 to 2000 Hz. (c) Groundtruth pitch states. In each time frame, the probability of a pitch state is 1 if it corresponds to the groundtruth pitch and 0 otherwise. (d) Probabilistic outputs from the DNN. (e) Probabilistic outputs from the RNN. (f) DNN based pitch contours. The circles denote the generated pitches, and solid lines denote the groundtruth pitch. (g) RNN based pitch contours.

[42] to connect those pitch states according to neural network outputs.

The Viterbi algorithm utilizes the likelihood and the transition probability to calculate the cost in order to generate an optimal sequence. The likelihood in each frame $P(\mathbf{x}_t|s_t^i)$ is proportional to the posterior probability divided by the prior $P(s^i)$:

$$p(\mathbf{x}_t|s_t^i) \propto \frac{P(s_t^i|\mathbf{x}_t)}{P(s^i)} \tag{5}$$

where $P(s_t^i|\mathbf{x}_t)$ is the output of a neural network. The prior $P(s^i)$ and the transition matrix are directly computed from the training data. Note that, since we train the DNN with both pitched and unpitched frames, the prior of the unpitched state $P(s^0)$ is usually much larger than that of each individual pitched state, resulting in the relatively small likelihood of the unpitched state, and the Viterbi algorithm may bias towards pitched states. Hence, we introduce a parameter $\alpha \in (0, 1]$ multiplying the prior of the unpitched state $P(s^0)$ to balance the ratio between the pitched and unpitched states, which is chosen from a development set. We should also mention that the output of the RNN is the posterior probability given an observation of a sequence rather than a single frame, which does not exactly satisfy the assumption of the HMM and the Viterbi algorithm, but we ignore this for simplicity.

The Viterbi algorithm outputs a sequence of pitch states for a sentence. We convert the sequence of pitch states to the sequence of frequencies and then use a 3-point moving average for smoothing to generate final pitch contours.

Fig. 1 illustrates pitch tracking results using the proposed methods. The example is a female utterance from the TIMIT corpus [43], “Readiness exercises are almost continuous”, mixed with factory noise in -5 dB SNR. Fig. 1(a) and (b) show the spectrograms of clean speech and noisy speech from 0 to 2000 Hz (for better clarity) respectively. Comparing Fig. 1(b) with Fig. 1(a), the harmonics are severely corrupted by noise, leading to a major difficulty in pitch tracking. Fig. 1(c) shows the groundtruth pitch states extracted from the clean speech using Praat [4]. As shown in the figure, Praat even makes a few doubling or halving pitch errors at around 160 ms and 280 ms, but since these errors are not serious, we do not correct them and still treat them as the groundtruth. The probabilistic outputs of the DNN and the RNN are shown in Figs. 1(d) and (e), respectively. Comparing to Fig. 1(c), the probabilities of the correct pitch states dominate in most time frames in both Figs. 1(d) and (e), demonstrating that the neural networks successfully predict pitch states from noisy speech. In some time frames (e.g., 100 ms to 120 ms), the RNN yields better probabilistic outputs than the DNN, because the RNN is able to better capture the temporal context and its outputs are smoother than those of the DNN. Figs. 1(f) and Figs. (g) show pitch contours after Viterbi decoding. In the figures, both the DNN and the RNN produce accurate pitch contours. A few errors occur from 260 ms to 280 ms due to severe interference.

V. Experimental Results

A. Corpus

We evaluate the performance for the proposed approach using the TIMIT database [43], [25]. The training set contains 250 utterances including 50 male speakers and 50 female speakers. The noises used in the training phase include babble noise from [19], factory noise, and high frequency radio noise from NOISEX-92 [38]. Each utterance is mixed with every noise type in three SNR levels: -5, 0, and 5 dB, therefore the training set includes $250 \times 3 \times 3 = 2250$ noisy sentences. The test set contains 50 utterances including 25 male speakers and 25 female speakers. All utterances and speakers are not seen in the training set. The noise types used in the test set include the three training noise types and six new noise types: cocktail-party noise, crowd playground noise, crowd music, traffic noise, wind noise, and rain noise [20]. We point out that although the three training noises are included in the test set, the noise recordings are cut from different segments. Each test utterance is mixed with each noise in six SNR levels of -10, -5, 0, 5, 10, and 20 dB. We also test pitch tracking for clean speech. The groundtruth pitch is extracted from clean speech using Praat [4]. In addition, we test the proposed approach using 20 utterances in the FDA evaluation database [2] where the groundtruth pitch contours were derived from laryngograph data.

We evaluate pitch tracking results in terms of two measurements: detection rate (DR) [21] and voicing decision error (VDE) [33]. DR is evaluated on voiced frames, where a pitch estimate is considered correct if the deviation of the estimated $F0$ is within 5% of the groundtruth $F0$, and VDE indicates the percentage of frames are misclassified in terms of voicing:

$$\mathrm{DR} = \frac{N_{0.05}}{N_p}, \mathrm{VDE} = \frac{N_{p\to n} + N_{n\to p}}{N} \quad (6)$$

Here, $N_{0.05}$ denotes the number of frames with pitch frequency deviation smaller 5% of the groundtruth frequency. $N_{p\to n}$ and $N_{n\to p}$ denote the number of frames misclassified as unpitched and pitched, respectively. N_p and N are the number of pitched frames in groundtruth and total frames in a sentence, respectively.

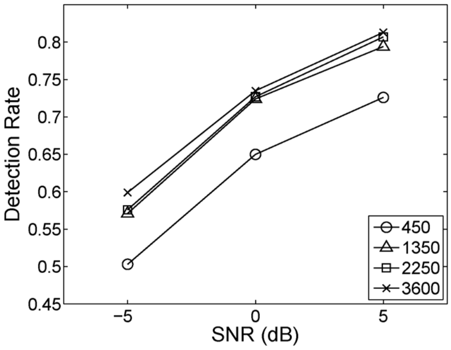

Fig. 2. Pitch detection rates of DNNs with different sizes of training set.

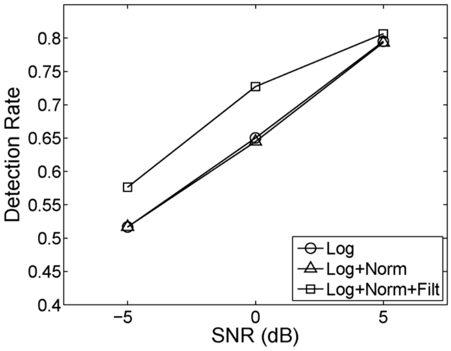

Fig. 3. Pitch detection rates of DNNs with different features. "Log" denotes PSD in the log-frequency domain, " $\mathrm{Log} + \mathrm{Norm}$ " denotes the normalized PSD, and " $\mathrm{Log} + \mathrm{Norm} + \mathrm{Filt}$ " denotes the filtered normalized PSD.

B. Parameter Selection

Since the proposed neural networks involve several parameters, we describe how to choose their values in this subsection. The size of training set influences on the performance, and we train four DNN models using different training sets with 450, 1350, 2250, and 3600 noisy sentences, corresponding to 50, 125, 250, 400 clean utterances. We compare the pitch tracking results in Fig. 2. The training set with 450 noisy sentences yields the lowest performance, while the other three produce rather comparable performances. In general, the performance increases with the increase of training set size, and the improvement becomes small when the size of training set reaches 1350 sentences.

Another important factor concerns features. In this study, we first compute the PSD $X(q)$ in the log-frequency domain, and then generate the normalized PSD $X'(q)$. The normalized spectral features are then convolved with a filter with a broadened impulse response, resulting the final features used in our study $\tilde{X}(q)$. To reveal feature effects, we train three DNN models using different features. As shown in Fig. 3, the filtered normalized PSD achieves the best performance, and the normalized PSD and the original PSD achieve comparable performance. The average detection rates are boosted by 5.0% by using the filtered normalized PSD.

We have conducted experiments for both DNN and RNN using different numbers of hidden layers. As shown in Fig. 4(a), the DNN with three hidden layers performs better than that with one hidden layer by 2.6% in detection rate and that with two hidden layers by 1.3%. As shown in Fig. 4(b), the RNN with two hidden layers produces comparable performance to that

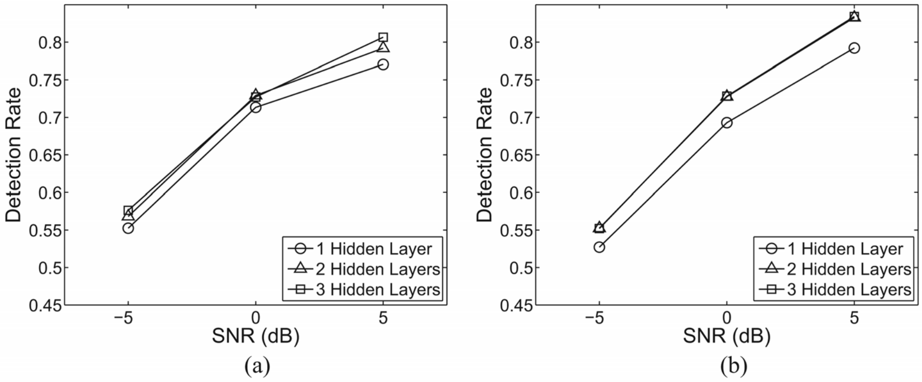

Fig. 4. Pitch detection rates with different numbers of hidden layers for (a) DNNs, and (b) RNNs.

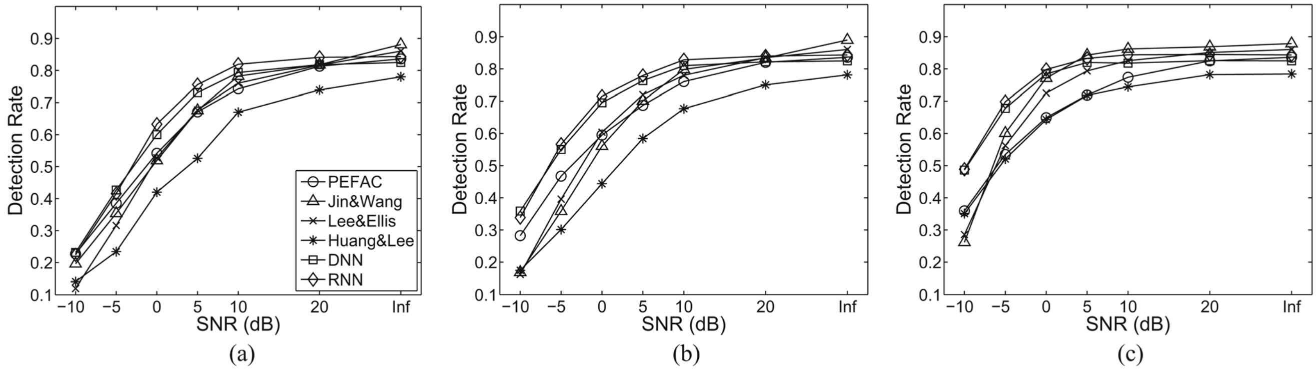

Fig. 5. Pitch detection rate comparisons for (a) babble noise, (b) factory noise, (c) high frequency radio noise.

with three hidden layers, but outperforms that with one hidden layer by 3.4%. We have also evaluated different numbers of hidden units, learning rates, and the numbers of neighboring frames. The parameter values used in this study are chosen using cross-validation from a development set.

C. Results and Comparisons

We compare our approach with four pitch tracking algorithms. PEFAC [11] extracts normalized spectral features to deal with strong noise and produces competitive pitch tracking results. The multipitch tracking algorithm of Jin and Wang [25] computes the autocorrelation function to select reliable channels and then utilizes an HMM to generate pitch contours (see also [42]). This algorithm is designed to handle reverberant noisy conditions. The third algorithm was proposed by Huang and Lee [22]. They compute a temporally accumulated peak spectrum as features and apply sparse reconstruction to estimate pitch in noise. The fourth algorithm was proposed by Lee and Ellis [28]. They extract subband autocorrelation and apply principal component analysis to reduce dimensionality. They train an MLP to estimate pitch. Note that, like ours the latter two algorithms require training and we use the same corpus (see Section V-A) to train these models for comparison.

Fig. 5 shows the detection rates for three training noises across a wide range of SNRs from -10 dB to clean (shown as "Inf" dB). The detection rates gradually increase with the increase of SNR. The DNN and the RNN based methods achieve substantially higher detection rates than others, especially in very low SNR conditions. The results of the unsupervised PEFAC algorithm are also notable, particularly for babble noise. Although we do not train the models under the SNR of -10 dB, the proposed approach still outperforms the others in this very low SNR condition. For the untrained high SNR conditions, the proposed approach also achieves good performance, although the relative advantage to others is not as large as in low SNRs. The proposed approach performs more than 5% better than all others on average and the advantage is more than 10% when the SNR is below 5 dB. The RNN performs slightly better than the DNN when the SNRs are greater than -5 dB.

Fig. 6 shows the detection rates for six new noises that are not seen in the training phase. Similar to Fig. 5, this figure shows that the proposed approach yields the best performance in these noise conditions, demonstrating that our supervised learning algorithms generalize well to different noisy environments. The average detection rates for the DNN and the RNN are 75% and 76% respectively, while the best comparison result is 72% for Lee and Ellis.

It is desirable for a pitch tracking algorithm to achieve high detection rates and low voicing detection errors at the same time. Since Huang and Lee's algorithm does not produce a pitched/unpitched decision, we only compare our approach with PEFAC, Jin and Wang, and Lee and Ellis. Fig. 7 and Fig. 8 show the VDE results for the seen and unseen noises, respectively. As shown in the figures, our algorithms produce lower voicing detection errors than others. On average, the VDEs of the DNN and the RNN based methods are both around 16% across all SNRs and noises, and the VDEs of PEFAC, Jin and

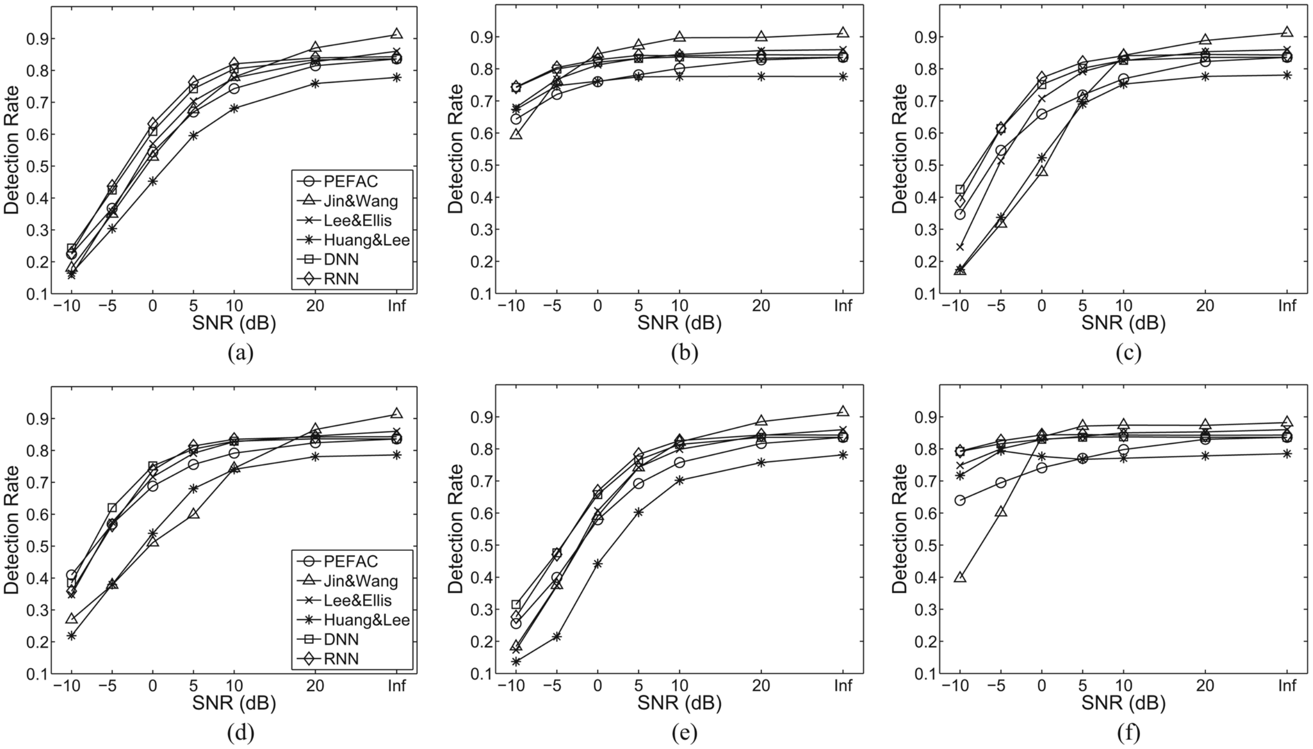

Fig. 6. Pitch detection rate comparisons for six new noises: (a) cocktail-party noise, (b) crowd playground noise, (c) crowd music, (d) traffic noise, (e) wind noise, (f) rain noise.

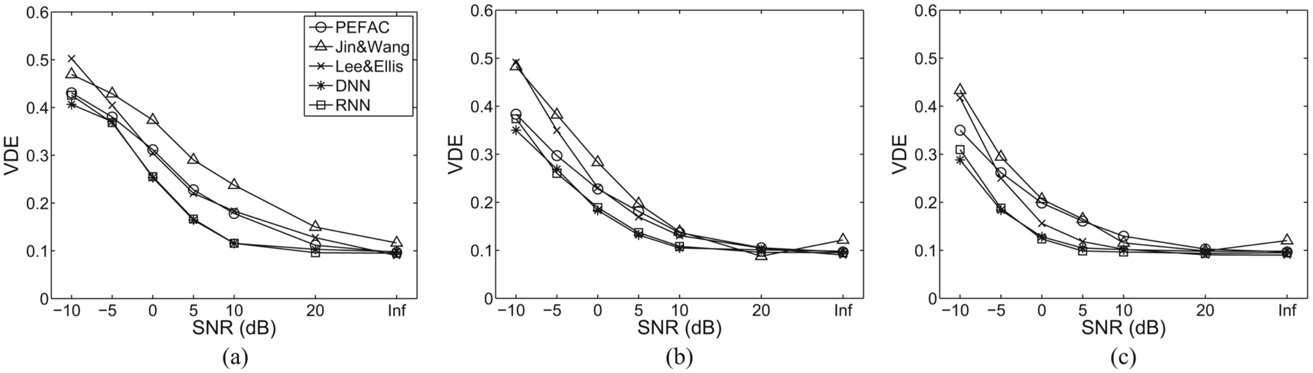

Fig. 7. Voicing detection error comparisons for (a) babble noise, (b) factory noise, (c) high frequency radio noise.

Wang, and Lee and Ellis are 20%, 25%, and 22%, respectively. The results indicate the superiority of the proposed approach on both pitch and voicing detection.

In the above experiments, the groundtruth pitch is extracted from clean speech using Praat, which is prone to some pitch detection errors. We now use the FDA database [2] to evaluate our approach without any retraining, where the groundtruth pitch is derived from laryngograph data. Fig. 9 shows the average pitch tracking results over three training noises and four different SNRs. The average detection rates for the DNN and the RNN are 51% and 50% respectively, which are higher than the others by around 6%. These and voicing detection results are consistent with those using Praat detected pitch as groundtruth.

In Eq. (6), the denominator of the detection rate is the number of all pitched frames in the groundtruth, so it counts false rejects as errors. Other studies [42], [33] used gross pitch error (GPE) to evaluate pitch deviation over 20% in the frames where both the groundtruth and a pitch estimator produce a pitch. We have also used GPE to compare the performances of different approaches in six SNR conditions. The DNN and the RNN achieve GPEs of 6.6% and 5.7%, respectively. Lee and Ellis also achieve GPE of 5.7%. All others have GPEs higher than 9%.

VDE aggregates false rejects and false alarms together. Specifically, false reject is the percent of unpitched frames in a reference sentence wrongly classified by an estimator, and false alarm is the percent of pitched frames wrongly classified. Looking at these two kinds of error separately, the DNN and the RNN achieve low false reject rates in low SNR levels, that is, they can correctly detect pitched frames even when noise is very strong. On average, the false reject rates for the DNN and the RNN are 12% and 10% respectively, and Jin and Wang achieve the next best at 15%. The false alarm rates for all methods are comparable, below 7% under most conditions.

We also conduct a computational complexity comparison among different approaches. We test 90 sentences with the total length of 270 s on a machine with Intel Xeon x5650 CPU, 8 GB

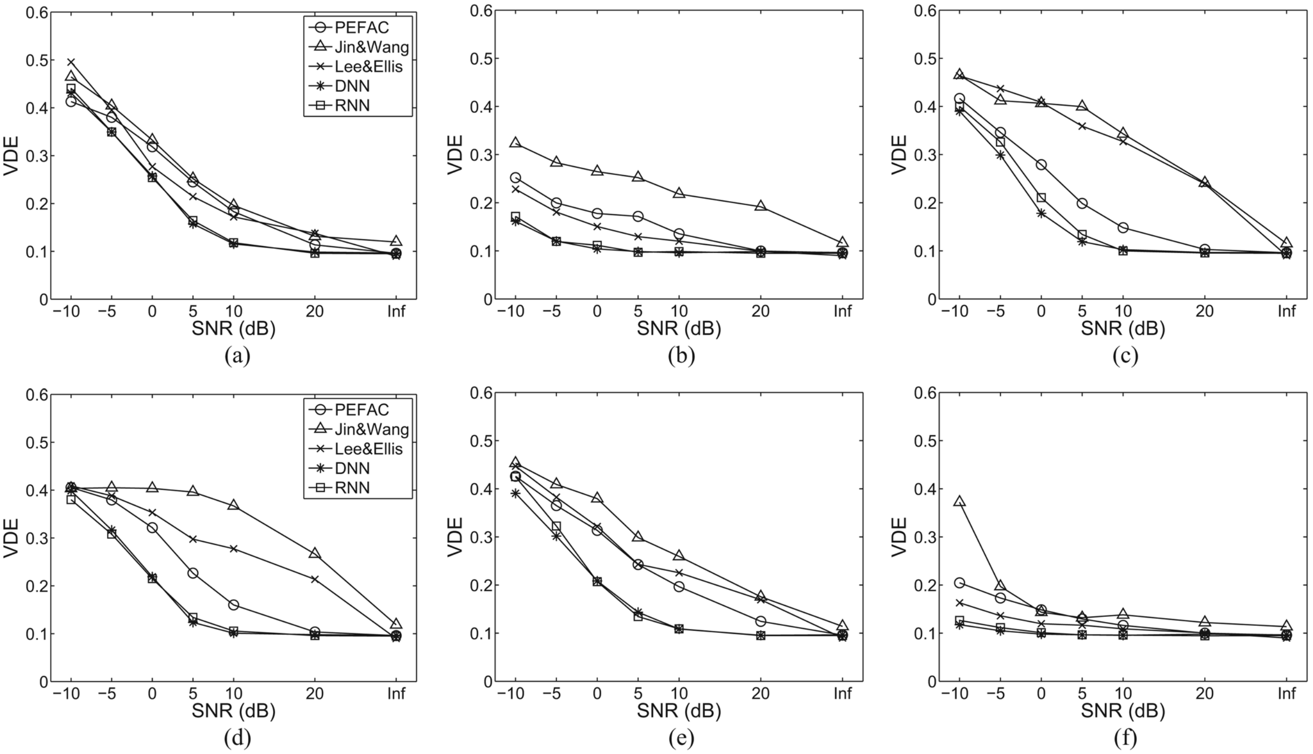

Fig. 8. Voicing detection error comparisons for six new noises: (a) cocktail-party noise, (b) crowd playground noise, (c) crowd music, (d) traffic noise, (e) wind noise, (f) rain noise.

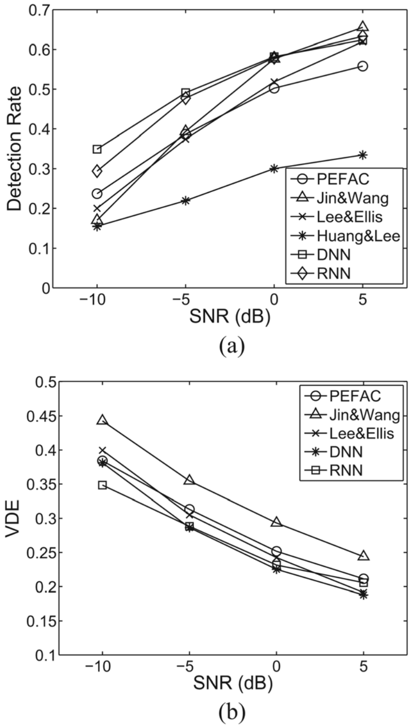

Fig. 9. Pitch tracking results for the FDA database. (a) Pitch detection rate. (b) Voicing detection error.

memory and NVIDIA M2070 GPU. Table I shows the average processing time per one second signal. Most approaches take less than 2 seconds, except for Jin and Wang which takes significantly more time.

TABLE I
RUNNING TIME COMPARISON FOR DIFFERENT APPROACHES

	PEFAC	Jin&Wang	Lee&Ellis	Huang&Lee	DNN	RNN
Time (s)	0.279	29.6	1.34	0.373	0.686	1.31

D. Extension to Reverberant Conditions

Reverberation smears the characteristics of harmonic structure and thus makes the task of pitch tracking more difficult. We apply the proposed approach to reverberant and noisy speech to evaluate the performance. In voiced speech, the fundamental frequency is defined as the rate of vibration of the vocal folds [26]. However, in reverberant conditions, the received speech is the filtered aggregated signal and the actual periodicity of the reverberant speech does not necessarily match its anechoic version. Because some speech processing applications would prefer a pitch estimate consistent with the harmonic structure of the reverberant speech [24], we consider the pitch of the reverberant speech as the groundtruth (see [25]).

Because the groundtruth of reverberant speech is different from that of anechoic speech, we need to retrain the models in reverberant conditions. To simulate room acoustics, we generate a simulated room corresponding to a specific reverberation time T_{60} [13] and randomly create a set of room impulse responses (RIRs) under this T_{60} condition. To train the system, we generate three reverberation times: 0.3, 0.6, and 0.9 s. The training set includes 250 utterances and three noises, both of which are the same as in the previous subsection. For each T_{60} condition, an utterance and a noise signal are convolved with two different RIRs respectively, corresponding to different source locations, and the two reverberant signals are then mixed at 0 dB SNR.

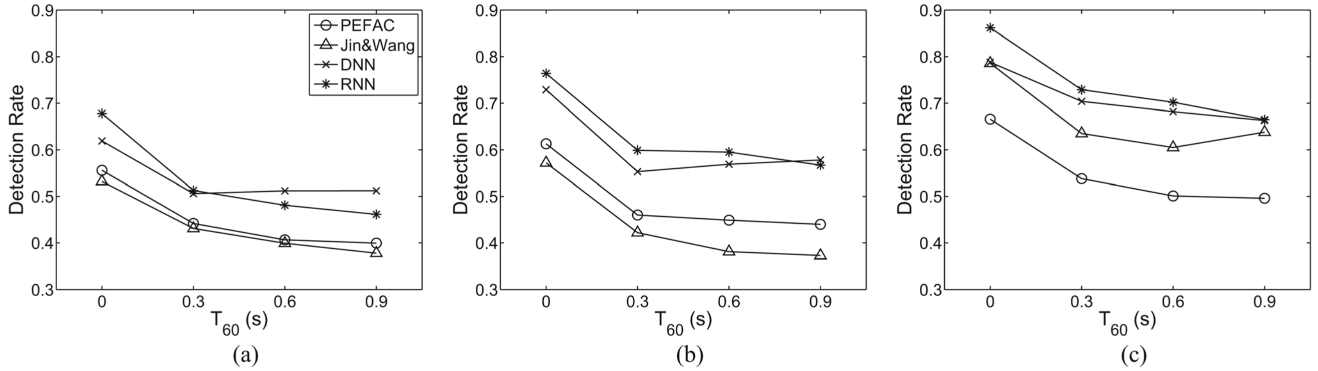

Fig. 10. Pitch detection rates for reverberant noisy speech: (a) babble noise, (b) factory noise, (c) high frequency radio noise.

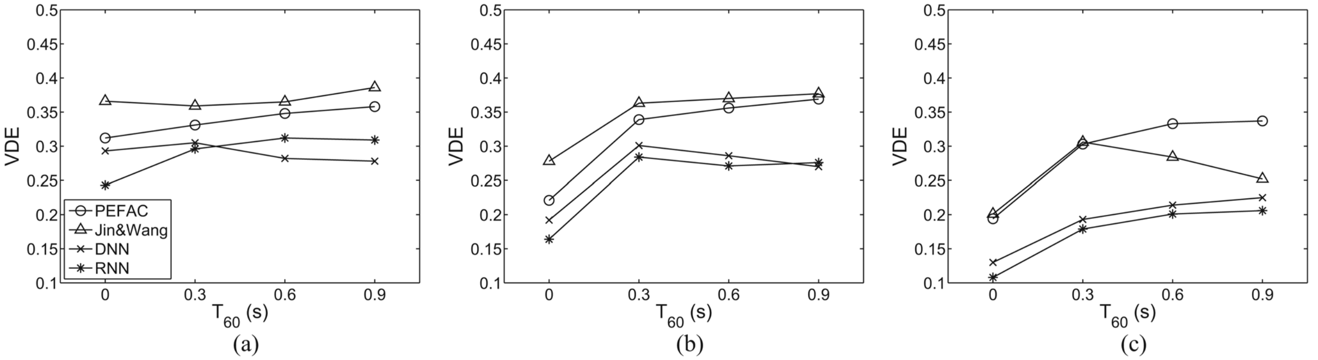

Fig. 11. Voicing detection errors for reverberant noisy speech: (a) babble noise, (b) factory noise, (c) high frequency radio noise.

Therefore, there are $250 \times 3 \times 3 = 2250$ reverberant sentences in the training set. The test set includes 450 sentences, consisting of 50 utterances mixed with the three training noises in three T_{60} s. Although the three T_{60} s are used in the training set, the RIRs in the test set are different from those in the training set. The groundtruth pitch is extracted from reverberant and noise-free utterances using Praat.

We compare our approach with PEFAC and Jin and Wang, because both have been shown to perform well in reverberation. In Fig. 10 and Fig. 11, we present the DR and the VDE results for reverberant and noisy speech with three T_{60} s and anechoic speech. As shown in the figures, although the performance for noisy reverberant speech is lower than that in the anechoic condition, the increase of the reverberation time starting from 0.3 s does not lead to significant performance degradation. We should point out that the anechoic condition is an unseen condition in this experiment, because the retrained model only uses reverberant speech. The fact that these results are broadly comparable to those in Fig. 5 and Fig. 7 at 0 dB indicates insensitivity of our supervised approach to reverberation. The proposed approach performs substantially better than PEFAC and Jin and Wang in terms of both detection rates and voicing detection errors. Here, the RNN outperforms the DNN except for the high T_{60} conditions in the babble noise.

The above experiments use Praat to extract pitch from reverberant, noise-free speech as the groundtruth. As done in the previous subsection, we evaluate the approaches using another pitch evaluation corpus [23] where reference pitch contours are labeled from reverberant speech by an interactive pitch determination algorithm [31], combining automatic pitch determination and human intervention. The original sentences in the corpus are randomly selected from the TIMIT corpus. Each anechoic sentence is convolved with RIRs in $T_{60} = 0.3$ s and $T_{60} = 0.6$ s, respectively (see [25] for details). We generate reverberant and noisy signals using babble, factory, and high frequency radio noises at 0 dB SNR, and obtain pitch tracking results without retraining.

Fig. 12 gives the pitch and voicing detection results of our approach and those of the comparison methods. As shown in the figure, both the DNN and the RNN based algorithms lead to significantly higher detection rates for all three noises. On average, the detection rates for the DNN and the RNN are 66.4% and 66.2%, respectively; while those of the others are all below 57%. In terms of voicing detection errors, the proposed approach achieves the lowest error rate on average. Broadly speaking, these results show similar trends to those in Figs. 10 and 11, and hence suggest that it is reasonable to use Praat to generate groundtruth pitch for training.

VI. Discussion

In this study, we use the supervised learning approach to learn the probability distribution of pitch states. Although supervised learning typically has a generalization issue, our system appears to exhibit very promising results across multiple unseen conditions, including different speakers, SNRs, noises, and room impulse responses. Some of previous supervised learning based pitch tracking algorithms perform well on trained conditions but need to be retrained in a new acoustic environment [5], [22]. We incorporate multiple conditions into a larger dataset and train a DNN or RNN model under different conditions, which potentially benefits the generalization ability of the system (see also [40]). The success of this multiple condition training is probably

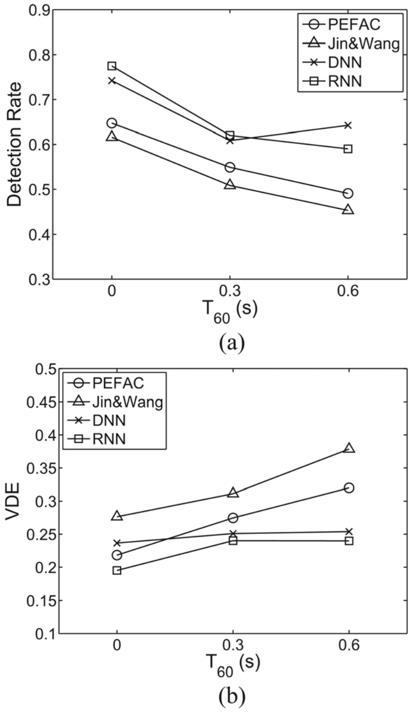

Fig. 12. Pitch tracking results on an interactively labeled pitch corpus: (a) detection rate, (b) voicing detection error.

due to extracted robust features as well as the learning capacity of the neural networks. We have tried to train single condition models for each specific acoustic environment, and found that single-condition training performs only slightly better than our multi-conditions training.

Our acoustic features for pitch estimation are computed from the filtered normalized log-frequency power spectrogram. The features use signal processing techniques to attenuate interference and facilitate subsequent neural network training. We have attempted to add an ACF based feature [28], but it only yields a slight improvement. In principle, the DNN is capable of learning high-level representation from raw data [17], [3] and recent advances in speech recognition [29], [7] also demonstrate that a DNN directly trained on the filter-bank features achieves better performance than trained on mel-frequency cepstral coefficient (MFCC) features. This suggests that, instead of using signal processing to generate features, we may consider raw features for neural network training in the future.

We have trained both DNN and RNN for pitch state estimation. Since post-processing can correct some pitch estimation errors from neural network outputs, the RNN does not produce significantly better results than the DNN in some conditions. However, the RNN intrinsically captures temporal dynamics, making it well suited for pitch tracking. As an example, Figs. 1(d) and (e) show the difference in pitch state estimation by the DNN and the RNN, and we can see that the output of the RNN is more smooth and continuous. In this study, we use the truncated BPTT to train the RNN and the longest time step is set to $T = 15$. A 15-frame truncation is not a long segment for pitch tracking, as the pitch contours in our study usually have 30 to 50 frames. We have tried to use 20-frame BPTT to train the models, but the results are similar, probably because training has reached a saturation point on our training dataset. Another strategy to train the RNN is to use BPTT on each sequence rather than a fixed-length segment. With sufficient training data the RNN is expected to encode longer dynamics, which may lead to performance improvement. In addition, we use a simple RNN in our study, and it is worth exploring other RNNs in future work, for example, long short term memory (LSTM) [18], which has demonstrated better performance than the simple RNN in some applications [12].

With neural network outputs, we use the Viterbi algorithm to generate pitch contours in the framework of HMMs. In other words, we assume that 1) the observation only depends on the hidden state in the current time step, and 2) the hidden state in the current time step only depends on the previous hidden state. To relax these assumptions, some studies use conditional random fields (CRFs) to model the sequence [27], [10]. We have attempted to use the CRF to generate the best sequence, but the performance is only slightly better than Viterbi decoding. It may be because the neural networks yield adequate information and a simple post-processing technique can achieve good results. Due to its complexity, we do not incorporate the CRF in our system, but it will be interesting to explore better sequence models for pitch tracking.

To conclude, we have proposed DNN and RNN to estimate the posterior probabilities of pitch states for pitch tracking in highly noisy speech. The supervised learning based approach produces strong pitch tracking results in both seen and unseen noisy conditions. In addition, the proposed approach can be extended to reverberant conditions.

ACKNOWLEDGMENT

We thank F. Huang and T. Lee for providing their pitch tracking code to us.

REFERENCES

[1] B. S. Atal, “Automatic speaker recognition based on pitch contours,” *J. Acoust. Soc. Amer.*, vol. 52, pp. 1687–1697, 1972.

[2] P. C. Bagshaw, S. M. Hiller, and M. A. Jack, “Enhanced Pitch Tracking and the Processing of F0 Contours for Computer Aided Intonation Teaching,” in *Proc. Eurospeech*, 1993, pp. 1003–1006.

[3] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[4] P. Boersma and D. Weenink, PRAAT: Doing Phonetics by Computer (version 4.5). 2007 [Online]. Available: http://www.fon.hum.uva.nl/praat

[5] W. Chu and A. Alwan, “SAFE: A statistical approach to F0 estimation under clean and noisy conditions,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 3, pp. 933–944, Mar. 2012.

[6] A. De Cheveigné and H. Kawahara, “YIN, a fundamental frequency estimator for speech and music,” *J. Acoust. Soc. Amer.*, vol. 111, p. 1917, 2002.

[7] L. Deng, J. Li, J.-T. Huang, K. Yao, D. Yu, F. Seide, M. Seltzer, G. Zweig, X. He, and J. Williams *et al.*, “Recent advances in deep learning for speech research at Microsoft,” in *Proc. ICASSP*, 2013, pp. 8604–8608.

[8] J. L. Elman, “Finding structure in time,” *Cognitive Sci.*, vol. 14, no. 2, pp. 179–211, 1990.

[9] G. D. Forney, Jr, “The Viterbi algorithm,” *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[10] E. Fosler-Lussier, Y. He, P. Jyothi, and R. Prabhavalkar, “Conditional random fields in speech, audio, and language processing,” *Proc. IEEE*, vol. 101, no. 5, pp. 1054–1075, May 2013.

[11] S. Gonzalez and M. Brookes, “PEFAC-A pitch estimation algorithm robust to high levels of noise,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 2, pp. 518–530, Feb. 2014.

[12] A. Graves, *Supervised sequence labelling with recurrent neural networks*. New York, NY, USA: Springer, 2012, vol. 385.

[13] E. Habets, Room Impulse Response Generator. 2010 [Online]. Available: http://home.tiscali.nl/ehabets/rirgenerator.html

[14] K. Han and D. L. Wang, “A classification based approach to speech segregation,” *J. Acoust. Soc. Amer.*, vol. 132, no. 5, pp. 3475–3483, 2012.

[15] K. Han and D. L. Wang, “Neural networks for supervised pitch tracking in noise,” in *Proc. ICASSP*, 2014, pp. 1488–1492.

[16] D. J. Hermes, “Measurement of pitch by subharmonic summation,” *J. Acoust. Soc. Amer.*, vol. 83, p. 257, 1988.

[17] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[18] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] G. Hu, 100 Nonspeech Sounds 2006 [Online]. Available: http://www.cse.ohio-state.edu/pnl/corpus/HuCorpus.html

[20] G. Hu, “Monaural speech organization and segregation,” Ph.D. dissertation, The Ohio State Univ. , Columbus, OH, USA, 2006.

[21] G. Hu and D. L. Wang, “A tandem algorithm for pitch estimation and voiced speech segregation,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 18, no. 8, pp. 2067–2079, Aug. 2010.

[22] F. Huang and T. Lee, “Pitch estimation in noisy speech using accumulated peak spectrum and sparse estimation technique,” *IEEE Trans. Speech, Audio Process.*, vol. 21, no. 3, pp. 99–109, Mar. 2013.

[23] Z. Jin and D. L. Wang, *Reverberant Pitch Evaluation Corpus*, 2011 [Online]. Available: http://www.cse.ohio-state.edu/pnl/shareware/jin1-taslp11/

[24] Z. Jin and D. L. Wang, “A supervised learning approach to monaural segregation of reverberant speech,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 17, no. 4, pp. 625–638, May 2009.

[25] Z. Jin and D. L. Wang, “HMM-based multipitch tracking for noisy and reverberant speech,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 5, pp. 1091–1102, Jul. 2011.

[26] V. Kalatzis and C. Petit, “The fundamental and medical impacts of recent progress in research on hereditary hearing loss,” *Human Molecular Genetics*, vol. 7, no. 10, pp. 1589–1597, 1998.

[27] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proc. ICML*, 2001, pp. 282–289.

[28] B. S. Lee and D. P. W. Ellis, “Noise robust pitch tracking by subband autocorrelation classification,” in *Proc. Interspeech*, 2012.

[29] J. Li, D. Yu, J.-T. Huang, and Y. Gong, “Improving wideband speech recognition using mixed-bandwidth training data in CD-DNN-HMM,” in *Proc. SLT*, 2012.

[30] A. L. Maas, Q. V. Le, T. M. O’Neil, O. Vinyals, P. Nguyen, and A. Ng, “Recurrent neural networks for noise reduction in robust ASR,” in *Proc. Interspeech*, 2012.

[31] C. McGonegal, L. Rabiner, and A. Rosenberg, “A semiautomatic pitch detector (SAPD),” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 23, no. 6, pp. 570–574, Dec. 1975.

[32] A. Mohamed, G. E. Dahl, and G. Hinton, “Acoustic modeling using deep belief networks,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 1, pp. 14–22, Jan. 2012.

[33] T. Nakatani, S. Amano, T. Irino, K. Ishizuka, and T. Kondo, “A method for fundamental frequency estimation and voicing decision: Application to infant utterances recorded in real acoustical environments,” *Speech Commun.*, vol. 50, no. 3, pp. 203–214, 2008.

[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[35] M. R. Schroeder, “Period histogram and product spectrum: New methods for fundamental-frequency measurement,” *J. Acoust. Soc. Amer.*, vol. 43, p. 829, 1968.

[36] I. Sutskever, “Training recurrent neural networks,” Ph.D. dissertation, Univ. of Toronto, Toronto, ON, Canada, 2013.

[37] D. Talkin, “A robust algorithm for pitch tracking (RAPT),” *Speech Coding Synth.*, vol. 495, p. 518, 1995.

[38] A. Varga and H. J. M. Steeneken, “Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems,” *Speech Commun.*, vol. 12, no. 3, pp. 247–251, 1993.

[39] O. Vinyals, S. V. Ravuri, and D. Povey, “Revisiting recurrent neural networks for robust ASR,” in *Proc. ICASSP*, 2012, pp. 4085–4088.

[40] Y. Wang and D. L. Wang, “Towards scaling up classification-based speech separation,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 21, no. 7, pp. 1381–1390, Jul. 2013.

[41] R. J. Williams and J. Peng, “An efficient gradient-based algorithm for on-line training of recurrent network trajectories,” *Neural Comput.*, vol. 2, no. 4, pp. 490–501, 1990.

[42] M. Wu, D. L. Wang, and G. J. Brown, “A multipitch tracking algorithm for noisy speech,” *IEEE Trans. Speech, Audio Process.*, vol. 11, no. 3, pp. 229–241, Mar. 2003.

[43] V. Zue, S. Seneff, and J. Glass, “Speech database development at MIT: TIMIT and beyond,” *Speech Commun.*, vol. 9, no. 4, pp. 351–356, 1990.

Kun Han received the Ph.D. degree from the Ohio State University in computer science in 2014 and the M. S. degree from the University of Science and Technology of China, Hefei, China, in 2008. His research interests include speech processing and machine learning.

DeLiang Wang, photograph and biography not provided at the time of publication.