Fundamental frequency estimation using deep neural networks

Alexander Kittel
Dept. of Health Technology, Technical University of Denmark
Supervisor: Tobias May
Dept. of Health Technology, Technical University of Denmark

Dec. 2021

Abstract

Pitch tracking is an integral aspect of both speech and music signal processing. Using short time autocorrelation, one of the main pitch estimation techniques, is a tried-and-true method in terms of detecting periodicity features. However, simply tracking the peaks in the autocorrelation function of a signal is prone to errors (hereunder "octave errors"), and performance becomes increasingly worse with the addition of noise. A deep neural network classifier can be trained directly on the autocorrelation function of a noisy input signal, bypassing the explicit peak identification process. By training this estimator on speech mixed with fluctuating noise, a more robust pitch tracking method can be attained. Using pitch tracking error percentage as a metric, several preprocessing choices are compared for a range of classifiers. This includes running the signal through a gammatone filterbank and operating on subbands, downsampling over-represented pitch values and reducing the feature dimensionality by discrete cosine transform. Only shallow networks are examined in the scope of this project, with only one classifier using two hidden layers and the rest using just one. Unfortunately, the implementation of the discrete cosine transform was unsuccessful, which is reflected in the tracking errors. Even so, the other estimators (both broadband and subband), display better performance over a wide range of noise levels than the baseline peak tracking method.

1 Introduction

Fundamental frequency estimation is a central aspect of speech detection algorithms, as well as playing a part in noise reduction, adaptive compression schemes or any audio processing that depends on identifying speech dominated sections of a signal. These techniques are also used in digital assistant software, such as Amazon's Alexa or Apple's Siri. One way of estimating the pitch of speech in a signal is by autocorrelating time segments at varying lags, and identifying the peaks corresponding to the most dominant periodicity [1].

Pitch detection becomes more challenging when noise is introduced, especially at low signal-to-noise ratios (SNRs), and further still when the noise is non-stationary. This project is motivated by potential real-world applications, meaning the speech is mixed with noise from the DEMAND database, which includes recordings from offices, cafes and public transportation.

The baseline pitch estimation algorithm used in this work is based on identifying peak values in short-time autocorrelation functions of the broadband input audio. This method has been studied extensively, producing good results in clean recordings. However, it may lead to "octave errors" and other issues, especially with the addition of fluctuating noise [1].

In this paper, instead of explicitly tracking peaks corresponding to pitch values, a deep neural network (DNN) classifier inspired by the SAcC developed in [2] is trained. The benefit of this method is the potential of high accuracy in a wide variety of situations. The training data is chosen to be highly varied both in terms of the clean speech, but also with regards to the noise and SNR values of the individual noisy speech mixes.

A comparison framework using the pitch tracking error (PTE) [2] as function of SNR for each estimator, as well as the posterior pitch value probability of each classifier allows the performance of multiple estimators to be compared.

The following section describes the autocorrelation method used as a baseline pitch estimator. The different data pre-processing techniques used are described in section 3, and the metrics used to compare their effects on the estimator models are described in section 4. The experimental setup is given in section 5, and section 6 discusses the results obtained.

2 The autocorrelation method

The autocorrelation method is well documented for both direct pitch tracking and for machine applications [1], [2]. The process is the same for both broadband and subband. The normalized autocorrelation function (ACF) is calculated for the broadband input audio signal

$x(n)$ (here a mix of speech and noise) at 10 ms lag increments:

$$A(t,\tau) = \frac{r(t,\tau)}{\sqrt{r(t,0)}\sqrt{r(t+\tau,0)}}, \quad (1)$$

with t indexing the analysis frame at lag τ, and

$$r(t,\tau) = \sum_{n=-N/2}^{N/2} x(t+n)x(t+n+\tau). \quad (2)$$

In the case of using a filterbank, the ACF is calculated for each individual subband of the signal. The frequency range of interest in this setup is 50 to 350 Hz, which at 16 kHz sampling rate translates to a maximum lag of 320 ms and 106 total lag values. The analysis window is 32 ms long.

The baseline method applies parabolic interpolation to the extremes of the ACF on a frame-by-frame basis to estimate the pitch value at a given time. The risk of octave-errors is mitigated by a subsequent median filter with window size 5. This technique acts as a baseline for comparison with the learning based methods described further below.

3 The DNN approach

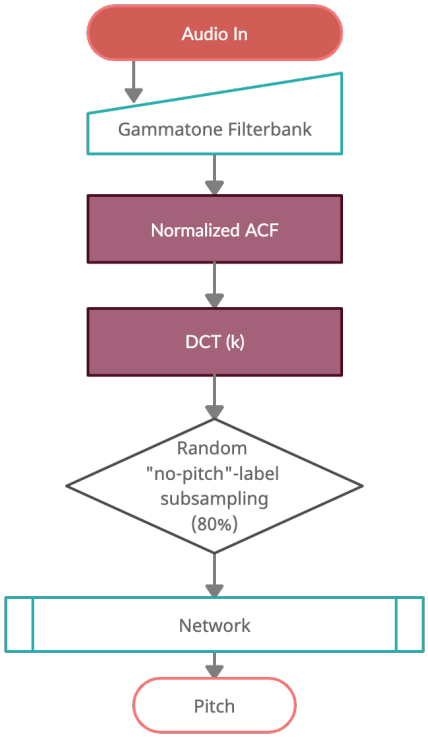

Fig. 1. Diagram of DNN-based frequency estimator. The general look is heavily inspired by [2], and can be modified in order to assess the effect of each step.

The diagram in Figure 1 represents the most complex version of the DNN based pitch trackers this project covers. The stages are based on [2], with the noted difference being the use of the discrete cosine transform (DCT) (see subsection 3.2) for dimensionality reduction purposes, as opposed to principal component analysis. The filterbank, DCT and random no-pitch subsampling (explained in subsection 3.1) are the main aspects of the model that are varied and omitted in order to compare performance across estimators.

The network itself is a simple classifier built using the MatLab Deep Learning Toolbox. The network uses ReLu activation and batch normalization. All but one model tested use a single hidden layer. A two hidden layer model is also trained, simply using duplicated ReLu + batchnorm layers. Each layer consists of 128 hidden units. The network outputs a label for each time frame, corresponding to one of 50 log-spaced frequency values in the range defined in the ACF calculation (50 to 350 Hz). Special labels are added: ”no-pitch” for unvoiced frames, ”too low” for values below 50 Hz and ”too high” for values above 350 Hz, giving us 53 output units in total.

3.1 Preparing data for learning

The data used for training, validating and testing the models consists of mixtures of clean speech sentences from the Pitch Tracking Database from Graz University of Technology (PTDB-TUG) [3]. This database consists of both male and female speakers with a variety of accents, and has the ground truth pitch for each recording. The full recording corpus consists of 4270 sentences, spoken by 10 men and 10 women. The speech is mixed with noise from the DEMAND database to mimic a wide variety of real-world scenarios [4]. In order to maintain independence between training, validation and test sets, each of these are created using different speakers and different sections of the noise files.

As most of the audio consist of mostly unvoiced ”no-pitch” frames, the class distribution was heavily skewed. To amend this, 80% of the unvoiced frames were removed at random from the training sets. After the ACF is computed, all datasets are normalized to unit variance and zero mean using weights calculated from the training set. The finalized input to the network has dimensions [nLags x 1 x nSubbands x nFrames], to be used with the imageInputLayer structure in the Deep Learning Toolbox.

3.2 Subband dimensionality reduction by DCT

With each subband, the amount of data is effectively doubled, leading to a massive feature space. This is cumbersome both in terms of generating the data, but the training time of the models is also heavily affected by this. In order to simplify the classification process, a discrete cosine transform matrix is calculated and applied to each subband of the signal, reducing the feature number from 106 lag values to k DCT components. The DCT matrices are calculated from the training set and applied to the validation and test sets subsequently. For all applications of DCT, k is set to 15 in order to strike a balance between slimming the feature space enough while still representing the original data adequately.

4 Pitch tracking performance metrics

Our central performance metric for the estimators is the pitch tracking error (PTE) percentage, as proposed by Byung Suk Lee, Daniel P. W. Ellis et. al. in [2] based on

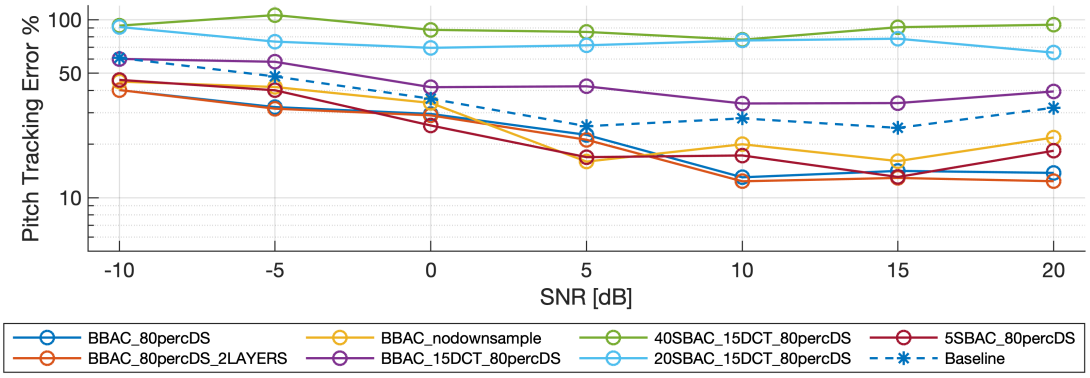

Fig. 2. Pitch tracking error percentage for all tested models as function of SNR. Each PTE value is averaged across several sentences and noise types for a given SNR. The best performers among the functional models are the broadband and "no-pitch"-downsampled estimators. The effect of a second hidden layer is a slightly (perhaps even negligibly) better PTE overall. Introducing a coarse filterbank seems to result in similar performance, though with some variation across SNRs. The same is valid for the model without downsampling, though it performs worse at the highest SNR values. The baseline method doesn't improve with SNR above 5 dB, which may be attributable to our decision to use non-stationary noise. Note: the three models using DCT should be disregarded as these values must be the result of some error in the data preprocessing. They are only included indicate that they are a work in progress at this point.

the work by M. J. Cheng, L. R. Rabiner, A. E. Rosenberg et al. in [5]. This metric averages voiced error (VE) and unvoiced error (UE), such that

$$\mathrm{PTE} = \frac{\mathrm{VE} + \mathrm{UE}}{2}, \tag{3}$$

with

$$\mathrm{VE} = \frac{E_{f_0} + E_{v\to u}}{N_v} \qquad \mathrm{UE} = \frac{E_{u\to v}}{N_u}. \tag{4}$$

N_v and N_u denote voiced and unvoiced frames respectively, E_{f_0} counts the number of frames where the estimate differs from the ground truth by more than 10%, and $E_{v\to u}$ and $E_{u\to v}$ denote misclassified unvoiced and voiced frames respectively. By using metrics that do not vary with the system (i.e. that may vary depending on our choice of frequency bin distribution and so on), estimators can be compared more transparently.

5 Experiment

5.1 Setup

The ground truth included in the PTDB-TUG is calculated using the same parameters as our models (32 ms window size with a 10 ms step size), which allows us to compare directly our estimations with the actual pitch. Since the output of all trained models is the same, they are compared using their pitch tracking error percentage values from -10 to 20 dB SNR. ACF inputs and pitch estimations are shown for randomly selected sentences at -10, 0 and 20 dB for the best performing model.

Each model is denoted BBAC or SBAC, which stands for broadband autocorrelation and subband autocorrelation respectively. SBAC is preceded by the number of subbands after the filterbank is applied. If DCT is applied, the number of components is also given in the title. Finally, the downsample percentage of "no-pitch" labels is given. For example, "20SBAC_15DCT_80percDS" indicates 20 subbands, 15 DCT components and a downsample rate of 80% of the "no-pitch" labels.

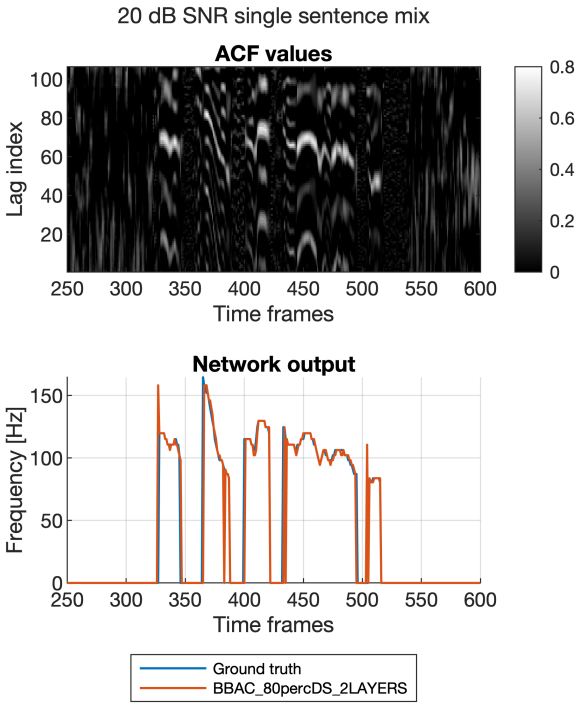

Fig. 3. Top: input ACF values for a randomly selected 20 dB SNR noisy speech mix, as function of both lag index and time frame. Bottom: network pitch classification as function of time frame. Note the lag index corresponds to lag times increasing downwards. This is done to display frequency increasing along the y-axis so that the structure of the ACF values more closely resembles the network output. Based on the PTE of this estimator, we expect good overlap between truth and estimated pitch at this SNR, which is what we observe for this particular sentence mix.

5.2 Results

The PTE as function of SNR is given for all tested models in Figure 2. The three worst performers are quite

obviously the result of some error in the application of the DCT, as they perform notably worse than even the baseline method. In the case of the 40 and 20 band models (green and cyan), the PTE approaches and even exceed 100%, regardless of SNR. These results are only included as the models were part of the objective, but they should be disregarded entirely. Further work is necessary to get these to function properly.

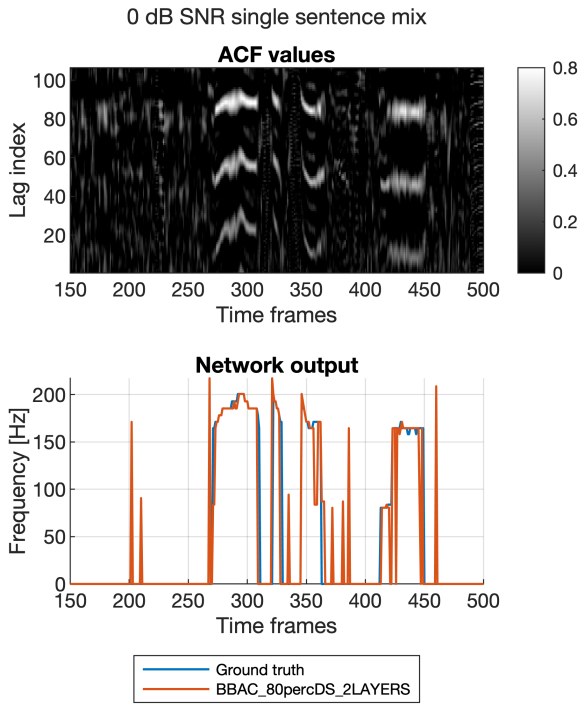

Fig. 4. Same as Figure 3 for a 0 dB SNR mix. There are more observable errors in the estimator, both in unvoiced and voiced frames. In particular for this scenario, there are a lot of individual unvoiced frame errors, as well as an example of octave error (the network estimates pitch at one octave below the ground truth) around frame 355.

Based on PTE percentage, the best estimator is the 2 hidden layer network using broadband ACF, and an 80% downsample of "no-pitch" labels. The input ACF and output pitch estimation for 20, 0 and -10 dB are given in Figures 3, 4 and 5 respectively. One of the major benefits of using a learning approach for pitch detection is apparent when comparing the PTE of the successful trained estimators to the baseline method. Above 5 dB SNR, the baseline method stops improving, whereas the trained estimators stabilize after 10 dB SNR. Results from [2] show that a functioning subband ACF classifier even improves up to 15 dB SNR, though their results are using pink noise.

6 Conclusion

By using fluctuating noise resembling speech (recorded in cafes, offices and traffic), trained estimators are shown to be much more robust than peak-tracking algorithms for varying noise scenarios. Training on varied speech using a wide variety of noise types is likely the biggest strength of this approach. We've shown good

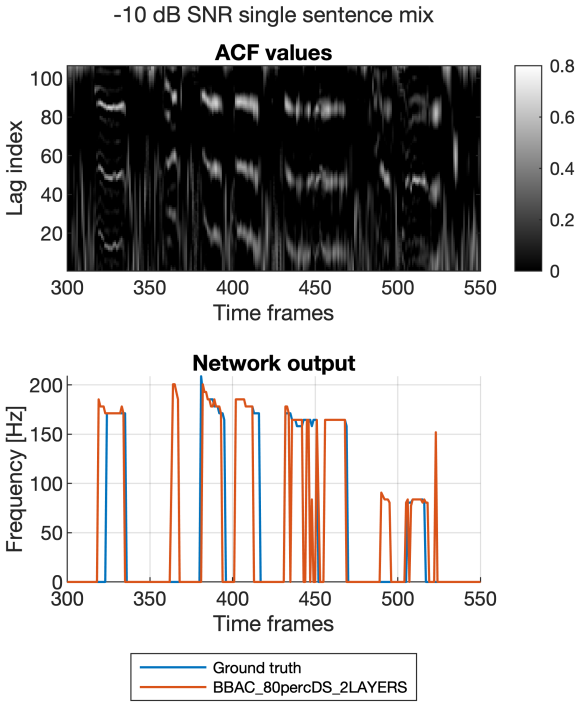

Fig. 5. Same as Figure 3. Several discrepancies can be observed between ground truth and estimated pitch. There seem to be some structures in the unvoiced frames due to the noise that the estimator is falsely classifying as voiced speech in this specific scenario. The estimator is also missing a lot of voiced frames. At this SNR we don't expect performance to be very good.

performance from a shallow network (two or fewer hidden layers) implementation of broadband ACF, using minimal data preparation. Downsampling the over-represented "no-pitch" label is also shown to increase performance for this choice of training set.

6.1 Discussion and further work

Several additional steps can be done to further increase performance, for instance (properly) applying dimensionality reduction to the feature space would enable the subband autocorrelation method without being hindered by a massive feature space. Furthermore, having the trained network output posterior pitch probabilities across the whole frequency range instead of single value classifications, and adding smoothing such as mean filtering or a Viterbi decoder [2] after the fact could improve results. This would be most apparent on the short time or single frame unvoiced errors, such as in Figure 4.

The choice of using prebuilt functions from the MatLab Deep Learning Toolbox was necessary for a fast and easy implementation, but there is a limit to the control offered by this medium, as well as a performance deficit compared to other tools. Translating this method to PyTorch, or another open-source deep learning library with more thorough control of model setup, may allow more parameters to be explored, proper parallel computing to be used and thus more complex models to be trained. In terms of network architecture, it could be

interesting to add a temporal context layer, which may further decrease single frame errors in estimation.

As was noted in the objectives, alternative periodicity features should be explored, such as cepstrum [**?**], harmonic product spectrum and more, in order to evaluate which feature set best translates to performance in a trained estimator. The evaluation framework proposed in this paper is expandable to other pitch estimation metrics, and would allow for these alternative models to be compared with ease.

REFERENCES

[1] M. Wu, D. Wang, and G. Brown, "A multipitch tracking algorithm for noisy speech," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 3, pp. 229–241, 2003.

[2] B. S. Lee and D. P. W. Ellis, "Noise robust pitch tracking by subband autocorrelation classification," 2012. [Online]. Available: http://www.icsi.berkeley.edu/Speech/qn.html

[3] G. Pirker, M. Wohlmayr, S. Petrik, and F. Pernkopf, "The pitch-tracking database from graz university of technology," 2012.

[4] J. Thiemann, N. Ito, and E. Vincent, "Demand: a collection of multi-channel recordings of acoustic noise in diverse environments," 2013.

[5] M. J. Cheng, L. R. Rabiner, A. E. Rosenberg, and C. A. McGonegal, "Comparative performance study of several pitch detection algorithms," *Journal of the Acoustical Society of America*, vol. 58, no. S1, pp. S61–S62, 1975.