White Noise Separation Using Convolutional Neural Network Architecture

Brian Yang[1]

Scarsdale High School, Scarsdale, New York, USA
e-mail: brian.a.yang@gmail.com
This work is under the supervision of Scarsdale High School Science Research and my adviser Guei-Yuan Lueh, PhD.

October 28, 2019

ABSTRACT

Speech enhancement is defined as a process that improves the quality of human voice in an environment with background noise. In this paper, we approach this subject with a variety of Artificial Intelligence techniques and a data loop-back feature. Our algorithm is comprised of a Convolutional Neural Network (CNN) that feeds into a transposed CNN, trained to separate background noise from the human voice. The architecture is designed to take *noise polluted voice recordings* as input and produce the same recordings with a high proportion of noise removed. The output of the network can be reprocessed as the input for further 'cleaning'. We used the TIMIT data set, mixed with background noise recordings to train our network. Our model is able to reduce white noise content in an audio sample by about 58% after one iteration, and by 89% after five iterations with a small distortion, approximately 15% to 20%, to the original voice.

Key words. CNN, Transposed CNN, noise reduction, TIMIT

1. Introduction

Background noise poses a significant challenge to acoustic speech recognition algorithms. Speech mixed with any white noises generally produces inaccurate results if fed through a speech-to-text neural network. The presence of noise may make it difficult for even a human being to discern what is said. Commercially available noise-cancelling headphones generally use a hardware device that measures ambient noise and produces a counter signal to cancel it *[NC (Wikipedia)]*. To actively cancel noise, the generated counter wave has to match both phase and amplitude of the noise wave; hence, it is difficult to use this technology in an open environment. Active noise-cancelling only functions effectively in headphones.

A widely successful mobile application **Shazam** can identify music titles even in a highly noisy environment. The application singles out a set of strongest peaks in each music title's spectrogram and uses the resulting scatter plot as the *finger print* for the title. The input is processed in a similar way: the input spectrogram is translated into a scatter plot representing the peaks in its spectrogram. Then a search between the input scatter plot and **Shazam** database is performed to find a match *[Wang (2003)]*. This technology does not contain any algorithm to reduce background noise.

In this research, we propose a deep neural network to reduce background noise for speech recorded in a noisy environment. The network we propose consists of two sub-networks. The first one is a CNN, and the second one is a *transposed* CNN (also known as *deconvolutional* network). This architecture, consisting of a CNN followed by a transposed CNN, is frequently used for image segmentation *[Noh, et al. (2015)]*.

2. Related Work

CNN has been the main architecture for computer vision related tasks. It has proven to be effective from image object classification to image segmentation.(*[LeCun, et al. (2015)]*,*[Zhang, et al. (2015)]*)

[Serc, et al. (2016)] experimented with various CNN structures for the purpose of Large Vocabulary Continuous Speech Recognition (LVCSR). In that study, a very deep CNN architecture with small 3x3 kernels and pooling layers embedded between convolution layers was tested in the context of LVCSR. A multi-scale feature map approach was taken, a process that is in emulated in our research. Sercu found the most successful architecture to be a very deep CNN with 14 weight layers and small 3x3 kernels. These findings demonstrate the viability of CNN for speech-related tasks.

[Huang, et al. (2014)] investigated the usage of deep learning models for monaural speech separation, employing a recurrent neural network for this task. The time series speech data is repeatedly passed through a single neural network along with previous output from the network. They were able to successfully perform the intended function and conclude that their recursive models can be used for many different applications. We will be implementing this recursive nature into our CNN.

[Lin, et al. (2018)] studied the process of separating a singing voice from its accompanying instrumental. They ap-

proached the challenge with a CNN, and trained it to produce an *IBM (Ideal Binary Mask)*. IBM serves as a matrix of binary masks, in which each value (either 0 or 1) indicates whether the corresponding spectrogram bin is part of voice or part of instrumental. Lin found much success with their CNN, as it was able to "compete with cutting-edge voice separation systems."

[Park Lee (2015)] compared the performance of Convolutional Neural Networks with Feedforward Neural Networks(FNN) and Recurrent Neural Networks(RNN) for speech enhancement. This study also used the TIMIT data set and combined its audio with various types of random background noise. Through their experimentation of the three types of networks, they found the CNN to perform much more efficiently than other network architectures. Their CNN was able to match the accuracy of the FNN and RNN, despite being many times smaller.

3. Data Pre-processing

Our input data starts as an audio file in the WAV format, all audio files are required to have a sampling rate of 16kHz. We use TIMIT data set as our audio source. TIMIT contains speech recordings of both male and female speakers from various regions in the US. Each recording comes with a *phoneme index* to mark the beginning and the end of every *phoneme* in the recording.

Every audio file is divided into an ordered sequence of *frames* with equal sizes. Each *frame* represents a 30ms recording, equivalent to 480 ($16K \times 0.03$) samples. The stride from one frame to the next is 10ms, hence adjacent frames are partially overlapping. A *Hann Window* function is applied to each frame to de-emphasize the boundaries of a frame. A Fast Fourier Transform is then performed on each frame, resulting in 240 frequency bands.

4. Architecture

The system consists of two neural networks in sequence. The first network has a similar architecture to that of a CNN-based classifier, but without the feed-forward layers at the end. Using a pre-processed audio frame and its context as input, it can extracts higher dimensional features. The resulting *feature map* is of much smaller size than the original input and can be used by a simple classification layer to classify an input into a phoneme. For the rest of the paper, this first network is referred to as the *feature extraction* network.

The input matrix to the feature extraction network is equivalent to a time slice from a spectrogram. Each element of the input matrix represents the intensity of a particular frequency channel. When attached to a fully connected network at the end, this network acts exactly as a frame-to-phoneme classifier. Inspired by the extensive usage of *inception* layers in GoogleLeNet [Szegedy, et al. (2015)], this network has two inception layers [Szegedy, et al. (2016)] at the top, followed by ten layers of conventional convolution layers. Layers with strides higher than one reduce the size of data after it is analyzed. The output from the last convolutional layer is the feature map, which is the input to the second network.

The second network is a *transposed CNN* [Dumoulin, et al. (2016)], aka *deconvolutional network* [Noh, et al. (2015)]. It takes the output of the feature extraction network, the feature map, as the input and transposes it into a *noise mask vector* $\overline{\mathbb{N}}$. The *noise mask vector* ends with the same size as the original input vector. The noise mask represents the noise components, by frequency, of a noise-polluted frame. For the rest of the paper, this transposed CNN is referred to as the *feature transpose* network.

With the exception of the last layer in the feature transpose network, which uses *hard tanh* function with range $[-1, 1]$ as the activation function, *LeakyReLU* is used through out the network.

Feature extraction network definition:

Type	Size	Stride	Filter Count
Inception	451	1	10
	1×45	1	10
	15×3	1	14
	5×5	1	14
Inception	25×1	1	10
	1×25	1	10
	5×5	1	40
Convolution	4×4	2	100
Convolution	3×3	1	100
Convolution	3×3	1	100
Convolution	3×3	1	100
Convolution	2×2	2	120
Convolution	3×3	1	120
Convolution	2×2	(1,2)	140
Convolution	2×2	(2,1)	160
Convolution	2×2	(1,2)	200

Feature transpose network definition:

Type	Size	Stride	Filter Count
Transposed Conv	2×2	(1,2)	160
Transposed Conv	2×2	(2,1)	140
Transposed Conv	2×2	(1,2)	120
Transposed Conv	3×3	1	120
Transposed Conv	2×2	2	100
Transposed Conv	3×3	1	100
Transposed Conv	3×3	1	100
Transposed Conv	3×3	1	100
Transposed Conv	4×4	2	60
Inception	25×1	1	48
	1×25	1	48
	5×5	1	48
Inception	45×1	1	1
	1×45	1	1
	15×3	1	1
	5×5	1	1

4.1. Operation Principle

A noise polluted spetrogram frame $\mathbb{F}$ is represented as $\mathbb{F} = \mathbb{V} + \mathbb{N}$ where $\mathbb{V}$ and $\mathbb{N}$ represents the voice and noise components respectively. The objective of the feature transpose network is to extract $\mathbb{N}$, the noise component, from a given $\mathbb{F}$.

A noise mask $\overline{\mathbb{N}}$, produced by the feature transpose network, consists of 240 values, in which each value represents the intensity of the nose component for a frequency band. The noise mask

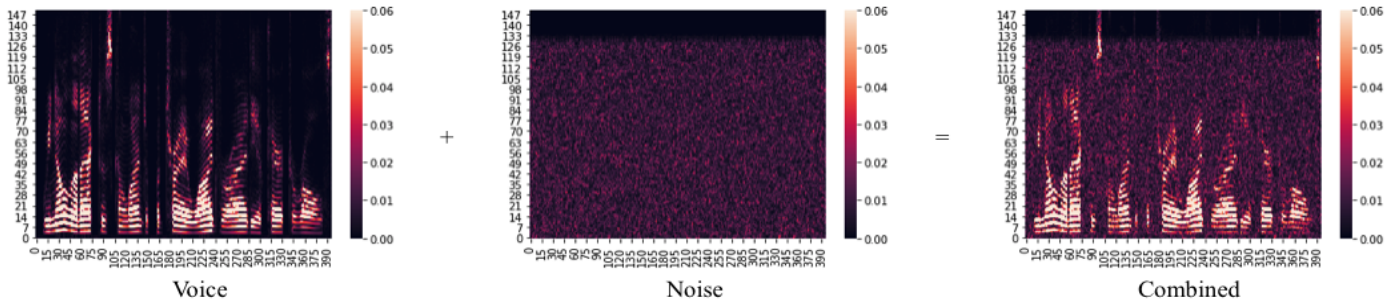

Fig. 1. Diagram illustrating the combining of white noise and voices.

may not represent the true noise $\mathbb{N}$ exactly. Subtracting $\overline{\mathbb{N}}$ from $\mathbb{F}$ may also degrade the quality of the voice component. Applying the mask $\overline{\mathbb{N}}$ to the original spectrum $\mathbb{F}$ can be represented as $\mathbb{F} - \overline{\mathbb{N}} = \overline{\mathbb{V}} + \mathbb{N}\mathrm{r}$, where $\overline{\mathbb{V}}$ is the resulting voice spectrum and $\mathbb{N}\mathrm{r}$ is the residual noise. A trained and well functioning network would need to exhibit the following two properties:

$(i)\ \overline{\mathbb{V}} \cong \mathbb{V}$ $\qquad$ $(ii)\ \|\mathbb{N}\mathrm{r}\| \lll \|\mathbb{N}\|$

The first property states that the voice component, after applying the noise mask, retains its quality. The second property states that the remaining voice component is much less prevalent.

4.2. Context-Sensitive Input

The objective of the feature extraction network is to classify each frame by its corresponding phoneme. Consecutive phonics are not mutually independent. Rather, the probability distribution of a phoneme at a particular time t depends on prior phonemes. In the field of *NLP (Natural Language Processing)*, given a sequence of words $x_0 ... x_{t-1} x_t$, the next word is frequently modeled by the following conditional probability: $P(x_{t+1}|x_t, x_{t-1}, \ldots x_0)$. Utilizing this information, many *NLP* networks are bidirectional recurrent networks, taking into time series data from both forward and backward directions.

In the research [Serc, et al. (2016)], a deep CNN network was proposed for LVCSR tasks. Deep CNNs with up to 14 layers were constructed and tested for LVCSR. The input to their networks contained three concatenated maps with different strides for the purpose of including contextual information.

Knowing that the context of a frame affect the probability distribution of a given frame, we include frames on both side of the input frame as part of the input data to the feature extration network. The input is formed by concatenating three (240×11) sub-matrices. All three are centered with the same input frame, making the center columns of all three sub-matrices identical while each one has a different stride for context frames. The first sub-matrix consists of the common center frame, along with five adjacent frames to the left and to the right of the center. The second sub-matrix has the same common center, but it takes every other frames to both sides until there are five frames on each side. The third sub-matrix, similarly constructed as the second one, but it takes every third frames to both sides. The end result is an input matrix by the size of $(3 \times 240 \times 11)$. Figure 2 illustrates the input construction.

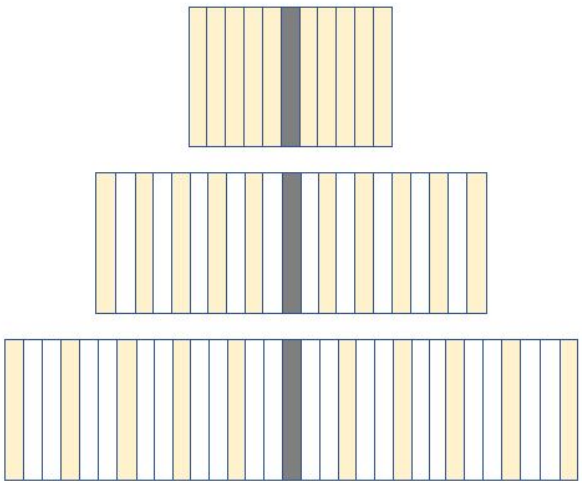

Fig. 2. Depiction of how the data is created to include context.

5. Training

The feature extraction network is first trained to classify the phoneme for a given audio sample. Only "clean" audio samples, i.e. samples without any background noise, are used to train this network. The entire TIMIT training dataset is used to train the feature extraction network. Once it is sufficiently trained, the last fully connected classification layers are discarded, and the feature transpose network is appended to the end. After this point, the weights of the feature extraction network remains fixed.

After connecting the two networks, they are trained with the goal of producing a noise mask $\overline{\mathbb{N}}$. The output of the network is trained to match the target of $\mathbb{F}$-$\mathbb{V}$, where $\mathbb{F}$ represents the noisy audio and $\mathbb{V}$ represents the clean voice. The loss function used here is *MSE (Mean Squared Error)*.

6. Experimental Results

To measure the quality of the cleaned audio frames and effectiveness of the network, we use the following three criterion:

(i) SNR (Signal Noise Ratio). This is the typical SNR used to measure the ratio between total voice energy and total noise energy. $SNR = \frac{\sum_{f,t} \mathbf{v}_{f,t}}{\sum_{f,t} \mathbf{n}_{f,t}}$, where $\mathbf{v}_{f,t}$ and $\mathbf{n}_{f,t}$ represent the voice, and noise energies, respectively

in a given spectrogram.

(ii) *NRR* (Noise Reduction Ratio). NRR is defined as the remaining noise energy in the cleaned frames as a percentage of total noise energy in the noisy recording. $NRR = \frac{\sum_{f,t} n_{f,t}}{\sum_{f,t} N_{f,t}}$, where $n_{f,t}$ and $N_{f,t}$ represent the noise energies for frequency f and time t bin of the cleaned, and noisy spectrograms, respectively.

(iii) *VDR* (Voice Distortion Ratio). This is defined as the total displaced voice energy as a ratio of total voice energy. It is calcuated as a root mean square. $VDR = \sqrt{\frac{\sum_{f,t}(v_{f,t} - V_{f,t})^2}{\sum_{f,t} V_{f,t}^2}}$, where $v_{f,t}$ and $V_{f,t}$ represent the voice energy for frequency f and time t bin of the cleaned, and the pure voice spectrograms, respectively.

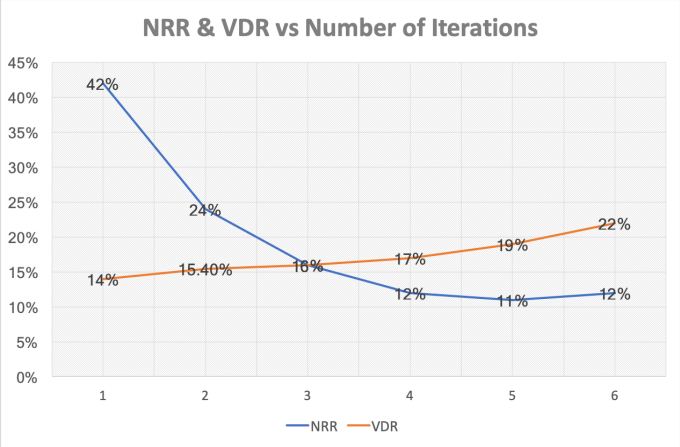

Fig. 3. NRR and VDR over iterations

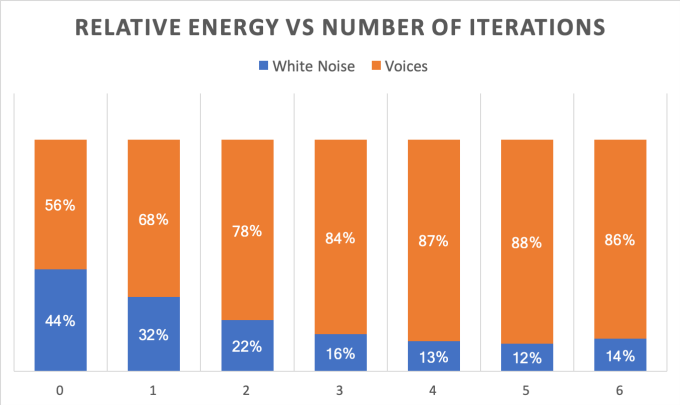

Fig. 4. Relative energy between voice and white noise over iterations. The 0th bar shows the original noise polluted recording.

6.1. Recursive Pass

The final output of this network $\mathbb{F} - \overline{\mathbb{N}}$ has exactly the same dimensions as the original input matrix. This allows the network to re-analyze the newly 'cleaned' result and produce a new noise mask. Such process is repeated for a number of times until noise can no longer be reduced or voice quality decades excessively.

Choosing a random combination of voice and white noise and running it through the network recursively, we found the results shown in Figure 3 and Figure 4.

The line representing the NRR, or the amount of noise present in the sample, loosely emulates the path of a logarithmic curve. The goal is to minimize the presence of noise, which means lower values of NRR are most desirable. The rate at which noise decreases starts at its maximum, and decreases as the sample is recursively passed through the network. In the test, the NRR reaches a minimum at five iterations, before increasing in value. The VDR, representing the distortion of the voice component in the resulting audio, follows a fairly linear path. Ideally, the voice remains unaltered, so lower values of VDR are most desirable. As the audio is iterated through the network, the VDR increases at a relatively constant rate.

The audio file started with SNR = 127%, indicating that 44% of the total sound was due to white noise and 56% was due to voice. As we continued the iterations, the voice component of the energy began to grow, representing a maximum of 88% of the energy in the audio. At this iteration the SNR was 708%, indicating the energy content was 7 times more than the white noise.

These results show the effectiveness of the network and strongly suggest that it satisfies the two critical properties that we set out to accomplish: First, $\overline{\mathbb{V}} \approxeq \mathbb{V}$, which states the voice component of the audio should remain relatively unaltered. VDR, or the amount of displaced voice energy, started at 14% after the first iteration, and reached a maximum of 22% after six iterations. Second, $||\mathbb{N}\mathrm{r}|| \lll ||\mathbb{N}||$, which states the noise in the audio should be greatly reduced. NRR, the remaining noise energy in the cleaned frame, more than halved to 42% after the first iteration and dropped to a minimum of 11% after five iterations.

7. Conclusions

In this paper, we combined various strategies relating to the processing of speech in order to test their viability in speech enhancement. We utilized a data-processing method, inspired by the work of *[Serc, et al. (2016)]*, that provides the network with context information for input sound samples. We used transposed CNN layers in the second portion of our network to upscale the data back to its original size. We also passed our data through the network multiple times, giving it a recursive nature.

In our randomly selected sample data, the energy of the noise was reduced by 58% after one iteration and by 89% after five iterations. The energy of the voice was altered by 14% after the first iteration and by 19% after five iterations. Although the network was able to significantly reduce the presence of the noise in the audio sample, the voice in the sample became more distorted as the data went through more iterations.

These findings could be a step towards more accurate text-to-speech algorithms, as the network increases the ratio of voice intensity to white noise. Our proposed CNN architecture performed fairly well, but some modifications may allow for more accurate results. Recursively analyzing data may be a

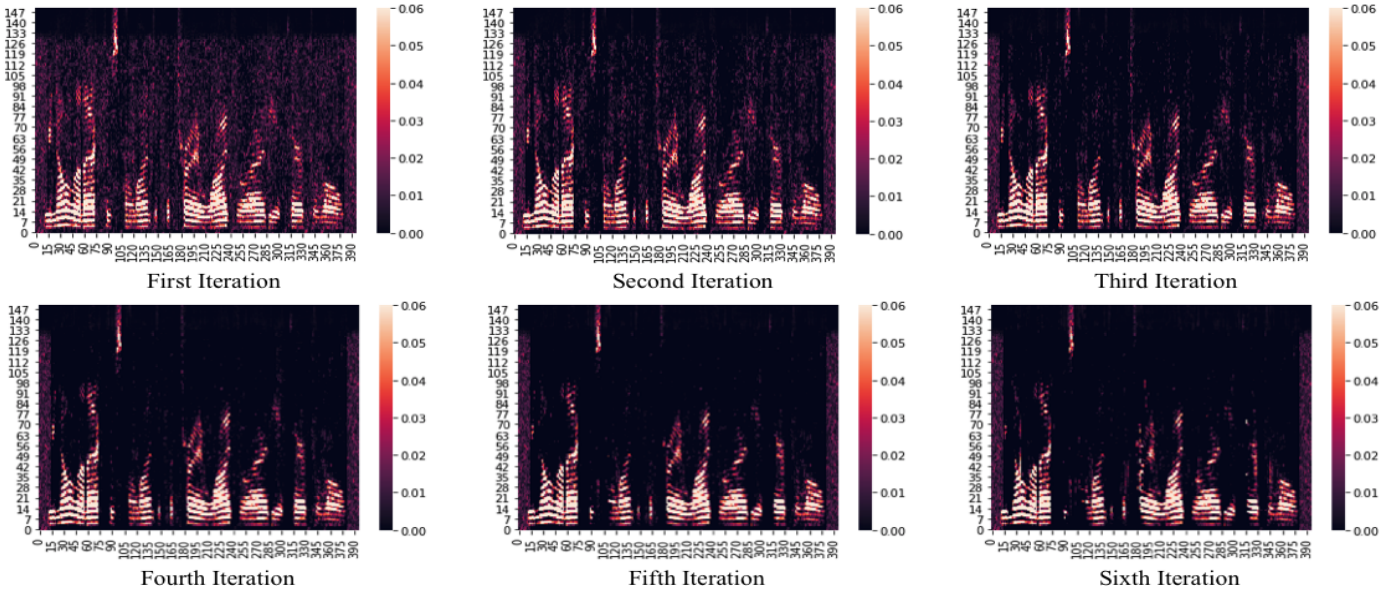

Fig. 5. Audio specgtrograms after each iteration through network. Note: 400 frames with the lower 150 (out of the 240) frequency bands are shown, higher frequencies are outside of human voice range. Color scales are normalized sound intensity.

methodology that could be implemented in the training process. In other words, the training of the 2nd network could be iterative by using the outputs as training data as well.

The specific dimensions and sizes of the various layers and kernels can also be experimented with, to possibly find a more efficient architecture. Future studies could also explore a network's accuracy when analyzing audio samples with different types of background noise, other than white noise.

References

Dumoulin, Visin 2016, "A guide to convolution arithmetic for deep learning", Université de Montréal

Huang, Kim, Hasegawa-Johnson, Smaragdis 2014, "Deep Learning For Monaural Speech Separation", 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

LeCun, Bengio, Hinton 2015, "Deep Learning", Nature 521

Lin, B.T., Lui, Herremans 2018, "Singing Voice Separation Using a Deep Convolutional Neural Network Trained by Ideal Binary Mask and Cross Entropy", Neural Computing and Applications

Noh, Hong, Han 2015, "Learning Deconvolution Network for Semantic Segmentation", 2015 IEEE International Conference on Computer Vision (ICCV)

Park, Lee 2015, "A Fully Convolutional Neural Network for Speech Enhancement", Interspeech 2017

Sercu, Puhrsch, Kingsbury, LeCun 2016, "Very Deep Multilingual Convolutional Neural Networks for LVCSR", 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

Szegedy, Vanhoucke, Ioffe, Shlens 2015, "Rethinking the Inception Architecture for ComputerVision", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Szegedy, Liu, Jia, Sermanet, Reed, Anguelov, Erhan, Vanhoucke, Rabinovich 2015, "Going Deeper With Convolutions", 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Wang 2003, "An Industrial-Strength Audio Search Algorithm", Shazam Entertainment, Ltd

Zhang, Zhao, LeCun 2015, "Character-level Convolutional Networks for Text Classification", Advances in Neural Information Processing Systems

Noise-cancelling headphones
"https://en.wikipedia.org/wiki/Noise-cancelling_headphones", Wikipedia