Speech recognition based on spectrograms and deep learning

Luis G. Leon-Vega
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
luis.leon@ieee.org

Allan Navarro-Brenes
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
allannabre01@estudiantec.cr

***Abstract*—Speech recognition has become one of the popular fields of machine learning that promotes a better user experience. This work proposes speech command recognition using Mel spectrograms as preprocessing step and Convolutional Neural Networks for classification, combining classical audio processing techniques and modern image classification techniques to carry out this task. Our work has presented an accuracy of more than 80% despite the simplicity against other methods such as autoencoders, recurrent neural networks and transformers.**

***Index Terms*—Deep Learning, Speech Recognition, Digital Signal Processing, Spectrograms, Digital Filters**

I. Introduction

Speech recognition is one of the most popular fields of research today, including popular applications such as speech commanding, and speech-based virtual assistants like Google Assistant, Cortana or Siri [ADD REFERENCE]. It comes with some advantages from the entertainment up to safety. In the safety case, for instance, car drivers can issue commands to their virtual assistants to dial another person, changing the song track, and interact with applications.

There are several approaches to addressing voice recognition in digital systems. One of the most popular before the deep learning era is the Hidden Markov Modelling (HMM), which is based on Markov chains, a stochastic model that describes a series of state transitions and their probability based on current observations. HMM, particularly, has hidden non-observable states which allow the system to learn [1]. The learning process is quite similar to other cutting-edge learning systems, where a stochastic process X (non-observable) is approximated by observing the output stochastic process Y (observable).

Deep learning is one of the cutting-edge techniques for voice and speech recognition. It is possible to find Recurrent Neural Networks (RNN) [2], Deep Neural Network (DNN) auto-encoders [3], and transformer-based networks [4]. These solutions can cope from single word recognition up to phrase recognition. Transformers are one of the most popular architectures for Natural Language Processing (NLP) nowadays, and they are not longer limited by text string length, removing limitations from other methods in languages like German.

The most recent research has focused on "end-to-end" speech recognition. Connectionist Temporal Classification is a type of neural network output and associated scoring function, for training RNNs to tackle sequence problems where the timing is variable [5].

This work focuses on single-command recognition. Using RNNs, auto-encoders or transformers are out-of-the-scope because they are over-engineered for the sake of this application. Therefore, this research will take advantage of classical Digital Signal Processing (DSP) techniques such as Mel filtering and spectrograms in combination with convolutional neural networks (CNN) for processing the resulting spectrograms from the DSP part.

II. Background

This work makes use of classical Digital Signal Processing techniques, such as the frequency domain analysis by using the Fourier Transformation, spectrograms, and Mel filtering. Moreover, it takes advantage of DNNs and CNNs for stochastic learning process, automatically tuning the parameters of the system to associate the spectrograms with a determined command through a latent space.

A. Frequency Domain Analysis

Analysing certain applications in the time domain can be complex in audio signal processing. For instance, the amplitude may change depending on how close the person is to the microphone and the length also depends on how fast the person speaks. These variances are not convenient for speech recognition because they add more degrees of freedom. However, converting the signal from the time-space to the frequency domain fixes the issue of speech speed and amplitude. Moreover, it constrains the signal within a sample length, analysing the frequency components of the incoming audio. It leads to a dimensional increase in the exchange for reducing the time dependency of a signal.

The Discrete Fourier Transform (DFT) [6] is one of the most popular tools for computing the frequency from a discrete signal. It is determined by the following equation:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{i2\pi}{N}kn} \tag{1}$$

where X_k is a frequency sample within the frequency domain and x_n is the sample from the signal in time-space.

The frequency in [Hz] of one of the frequency samples can be determined by

$$f = \frac{f_s \cdot k}{2N} \tag{2}$$

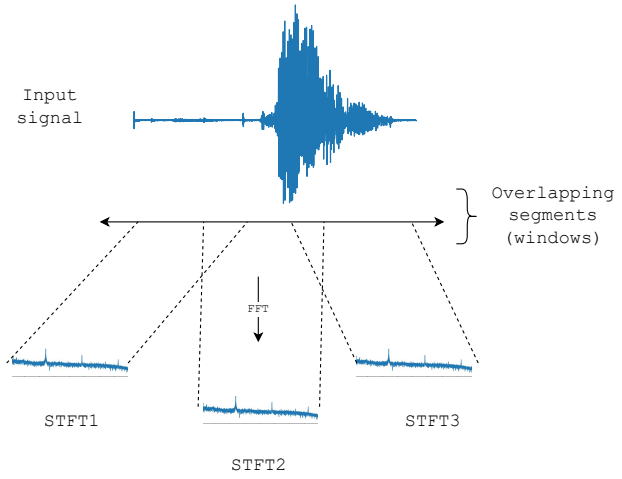

Fig. 1. Window-based frequency analysis

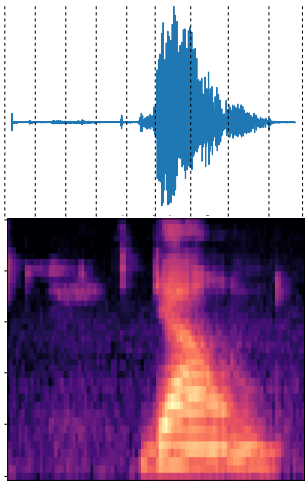

Fig. 2. Spectrogram illustration

where f is the hertzian frequency, f_s is the sampling frequency, k is the DFT sample, N is the number of samples, and the factor of 2 comes from the Nyquist sampling rate.

Hence, the DFT will produce a number of frequency samples equal to the number of samples of the input signal in time. Audio signals are finite energy signals with large lengths, which makes the DFT application inconvenient. In order to overcome this issue, it is possible to split the input signal into windows (which can be overlapping or non-overlapping windows) and compute the DFT over these signals. This is the key idea of the Short-term Fourier Transform (STFT) [7].

The STFT computes the DFT of each window in relation to a time framework. It means that it computes the energy of a signal within a time window (See Fig. 1). The concatenation of each STFT frequency window can lead to a *spectrogram*, relating frequency and time in a single plot [8].

Fig. 2 shows an example of a spectrogram. In this case, the signal is windowed into slices (overlapping or not) and each slice is analysed through the FFT. Then, the resulting FFTs are concatenated according to the offset of each slice in time. It is possible to compute the spectrogram through the equation (3)

$$\text{spectrogram}\{x(t)\}(\tau,\omega) \equiv |X(\tau,\omega)|^2 \tag{3}$$

where $x(t)$ is the signal in time, τ is the slow time (or time index), and ω is the angular frequency.

B. Mel scale

The Mel scale is a representation that maps frequency bands into a scale friendly to human perception [9], mimicking how humans perceive sound. The Mel filter-bank converts the hertzian frequencies into Mel bands through equation (4) [10].

$$m = 2595 \log_{10}(1 + \frac{f}{700}) \tag{4}$$

The effect of applying a Mel filter-bank that operates with equation (4) leads to a logarithmic mapping, as illustrated in Fig. 3.

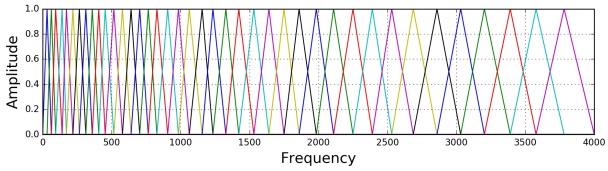

Fig. 3. Mel filter bank representation. It compress low frequencies and expands high frequencies in a logarithmic basis.

The resulting filter can be computed through

$$H_m(k) = \begin{cases} 0 & k < f(m-1) \\ \frac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \leq k \leq f(m) \\ \frac{f(m+1) - k}{f(m+1) - f(m)} & f(m) \leq k \leq f(m+1) \\ 0 & k > f(m+1) \end{cases}$$

Some applications discard the filter bank coefficients computed above because of their correlations, finding the Mel-frequency Cepstral Coefficients (MFCCs) as an alternative to decorrelate them. It is usually by applying the Discrete Cosine Transform (DCT) to the filter bank and yields to a compressed representation. Having a correlation within the inputs in machine learning algorithms is known to cause issues of biasing and this technique's results are convenient. For the sake of this work, the libraries to compute the Mel filter bank already take care of this issue.

C. Deep Learning Concepts

Neural networks have become one of the last decade's most important machine learning methods. They are competent against classic statistical methods in solutions and computational performance [11]. McCulloc and Pitts started with the initial foundation of an artificial neuron (also known as perceptron) based on the human brain [12]. An artificial neuron can be seen as a linear combination of multiple stimuli feeding an activation function, which is often non-linear:

$$y = \sigma(\mathbf{w} \odot \mathbf{x} + b) \tag{5}$$

where $\mathbf{x}$ is the input vector, $\mathbf{w}$ is the parameters' vector (also known as weights), b is the bias of the neuron,

$\sigma : \mathbb{R} \to \mathbb{R}$ is the activation function, and y is the output. $\odot$ is the dot-product between two vectors. Figure 4 (a) illustrates how a perceptron is graphically represented.

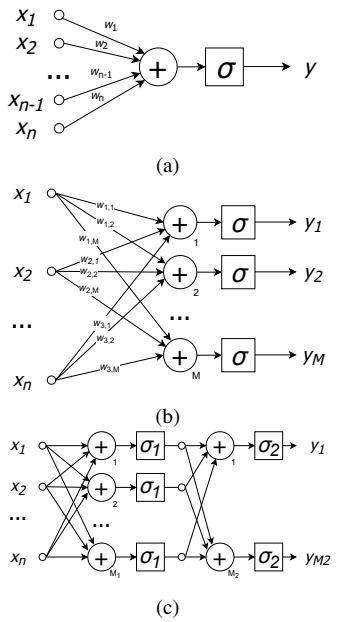

Fig. 4. Graphical representation of a neural network. **(a)** single perceptron, **(b)** multiperceptron layer, and **(c)** multi-layer perceptron

Rosenblatt trained a perceptron to solve linear regression problems in 1958 [13]. Later, Minsky and Papert (1969) will spot that a single perceptron cannot deal with the XOR function [14]. It motivated the exploration of multi-perceptron layers, also introduced by Rosenblatt. Then, the multilayer perceptrons arrived by cascading multiperceptron layers as in Figure 4 (c). In this former case, a network with multiple layers is also named as *deep neural network*.

D. Convolutional Neural Networks

Convolutional neural networks (CNNs) are a type of neural network widely adopted for pattern recognition in computer vision. CNNs deal with the computational complexity of computing dense neural networks for images due to their number of inputs (pixels). Their inspiration comes from the research on the visual cortex of a cat [15]. It describes how the neurons activate according to a stimulus coming from a screen. Then, the research was applied to pattern recognition of shift and deformation variant objects [16] and ML by using LeNet for classifying hand-written digits [17], introducing the concept of convolutional layers to artificial neural networks.

The relevance in neural networks relies on the compression of the trainable parameters, allowing NNs to learn features and compress them in feature maps (also known as convolution kernels) activated by a determined pattern when screening an entire image. The convolution operation between an image and a kernel is described by a Hadamard product and a sum reduction as an inner product in 2D per output pixel

$$Y_{i,j} = \sum_{n=-\kappa}^{\kappa} \sum_{m=-\kappa}^{\kappa} X_{i+n,j+m} K_{m,n} \qquad (6)$$

where Y_{ij} is the pixel at the i-th row and j-th column of the output $\mathbf{Y}$, $X_{i,j}$ is the pixel at the i-th row and j-th column of the input $\mathbf{X}$, and $K_{i,j}$ is the pixel at i-th row and j-th of the feature map $\mathbf{K}$ of size $(2\kappa + 1) \times (2\kappa + 1)$. The former equation describes the convolution in the spatial domain.

III. Speech recognition with Mel Spectrograms and Deep Learning

This work proposes an implementation for word recognition and classification by combining classical audio processing techniques in the frequency domain and deep learning. Our proposal consists of producing spectrograms in the Mel scale to illustrate the human perception of the voice and then processing the spectrograms through a CNN. The use of CNNs in audio processing is counter-intuitive and interesting. The idea is to represent the audio signal through the spectrogram using image analysis techniques using CNNs.

Fig. 5 shows a system overview of the proposal presented in this work. The preprocessing part utilises the STFT and Mel band transformation to compute a visual representation of the sound in the frequency domain (spectrogram). The spectrogram is fed into a CNN model to perform a sample classification task.

A. Filtering

Considering that a CNN model performs the classification task, it is required to feed the model with a 2D input. Considering the variance in time and amplitude from an audio signal in the time domain, the frequency analysis keeps a constant framework regarding power, isolating noise frequencies caused by the background and receiver from the actual signal power.

To carry out this task, we propose using audio spectrograms, a 2D representation of the signal, encapsulating time and frequency as the 2D axis and the signal power as the point value. The spectrograms are generated through the STFT computation, which does not have the inconvenience of requiring computing the entire DFT for the entire signal and can compute fragments of it, avoiding compromises in the frequency domain resolution.

Another important aspect is voice components. The bandwidth of the voice is from 300 Hz to 3400 Hz [18]. Hence, frequencies below 300 Hz are not relevant, and the analysis can neglect them. Nevertheless, the frequencies within the voice bandwidth are relevant to our work. Therefore, this work proposes using Mel frequency bands to provide a better expansion on the frequencies on the interest, compressing low frequencies which do not provide actual information about the human voice. This maximises the information transmitted to the CNN through the spectrograms.

To produce the Mel representation, we use the `librosa` [19] Mel spectrogram tool, which produces a Mel bank for the transformation of the raw spectrogram provided by the STFT to the Mel band scale. Afterwards, the resulting Mel spectrogram is then transformed to a logarithm scale for better compression.

Table I synthesises the configuration utilised to complete the processing using `librosa`.

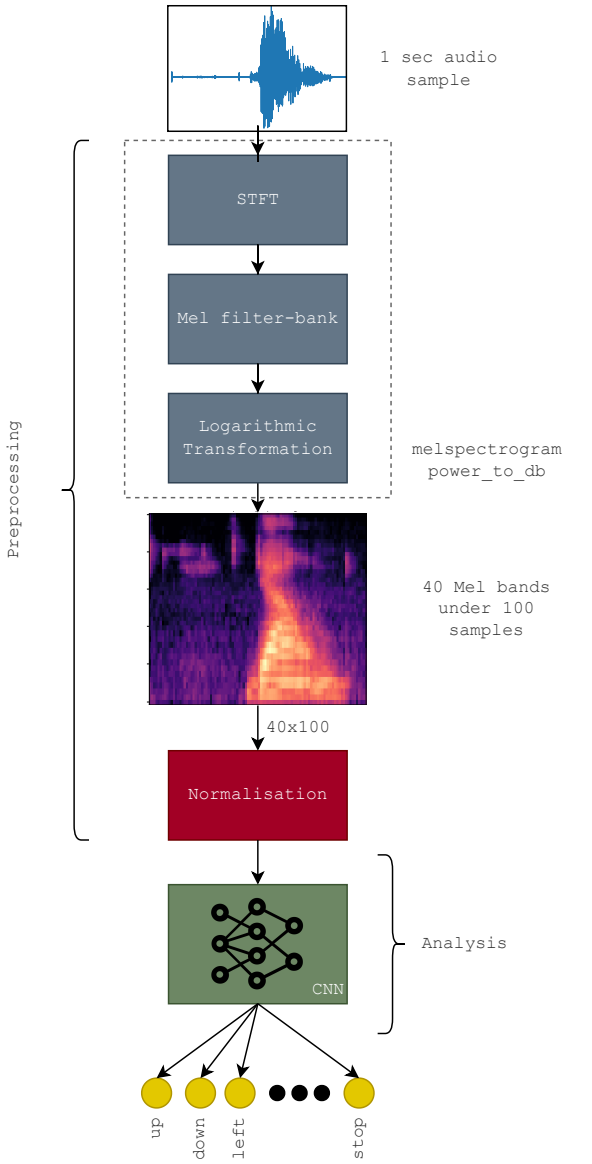

Fig. 5. System overview. The preprocessing stage utilises audio signal processing using the frequency domain and a Mel filter bank. The analysis receives the spectrogram as an image and continues a flow similar to an image analysis process.

TABLE I
PREPROCESSING CONFIGURATION

Property	Value
Sample duration	1s
Channels	1
Depth	16 bits
Sampling rate	16 kHz
Column/window duration	25 ms
Column distance (hop distance)	10 ms
Mel bands	40
FFT samples	8000
Signal spectrum output	Power in dB

TABLE II
NEURAL NETWORK TRAINING CONFIGURATIONS

Property	Value
Optimiser	Adam
Loss function	Cross Entropy
Learning rate	0.0003
Epochs	25
Batch size	128
Epoch shuffle	Yes

B. Convolutional Neural Network

Given that the preprocessing stage provides a 2D representation of the spectrum in the function of the time, we have found it convenient to use a CNN and proceed with a workflow similar to an image classification task.

The proposed model is illustrated in Fig. 6. The convolution layers allow feature extraction through kernels or maps. In the DSP context, the convolution operation triggers the outputs according to the similarity between the map (actually a filter) and the input signal, similar to the correlation operation.

The convolution layers are combined with subsampling through the `max` operation to reduce the dimensionality and batch normalisation to keep the probability distribution normalised. After the feature extraction steps precede a dense layer, which is in charge of weighting the contributions of each feature to the output classes, which are the desired commands, i.e. *up*, *down*, *left*, ..., *stop*.

Another layer is the *dropout*, which does not interfere during the inference but plays a role in the convergence in the training step, adding randomness to the process by disconnecting neurons and adding stochasticity to the backpropagation step.

For training purposes, our work proposes the configurations in Table II.

IV. RESULTS AND ANALYSIS

Figure 7 shows the distribution of labels for all the classes for training and validation. As shown in the figure, all categories have a similar quantity of training and validation samples the last ones representing around 12% of all samples.

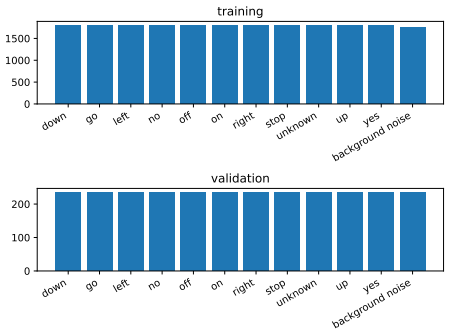

Fig. 7. Label distribution

The CNN model was trained using the Keras framework, the statistics for each batch are shown in Figure 8. The loss value drops significantly on the first batches and it converges to a value of around 0.02. As for the

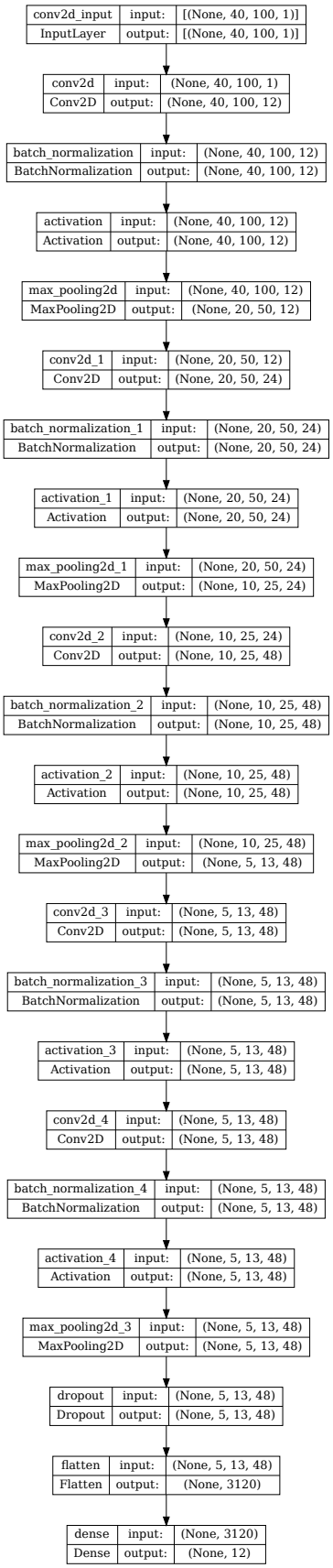

Fig. 6. CNN model utilised for the sample classification. The architecture is inspired by image classification task

accuracy, it increases during the early stage of the traing until reaching a value of approximately 0.992.

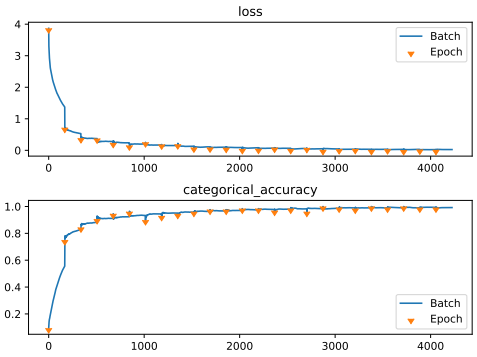

Fig. 8. CNN training statistics

A way to measure how fit a model is by using the confusion matrix. By using the model to predict the results using the validation dataset and determining the number of misses. Figure 9 shows that matrix for our model. The rightmost box shows the precision and false discovery rate, whereas the box at the bottom shows the recall and false negative rate.

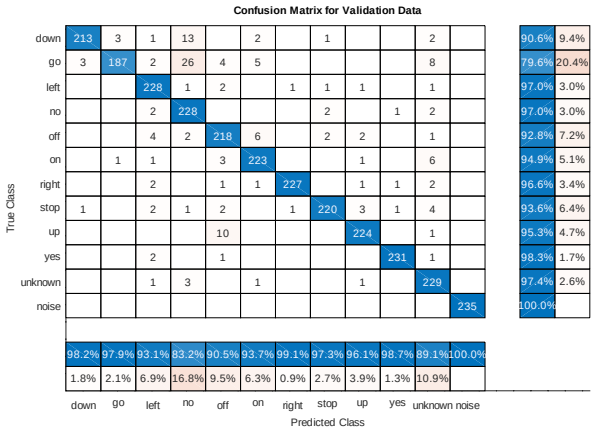

Fig. 9. Confusion matrix

Table III shows the results for audio clips recorded for testing, and which are not part of the original training or validation set.

TABLE III
PREDICTION RESULTS FOR RECORDED SAMPLES

File	result	Accuracy
off.wav	off	100.000%
left.wav	left	99.998%
go.wav	go	100.000%
right.wav	right	99.998%
push.wav	unknown	99.995%
no.wav	no	100.000%
yes.wav	yes	100.000%
up.wav	up	96.885%
down.wav	down	99.996%
hola.wav	unknown	97.442%
on.wav	on	100.000%
stop.wav	stop	100.000%

Table IV shows the performance of the CNN when calculation its CPU utilisation and the time it takes to process an spectogram.

TABLE IV
CNN PERFORMANCE

Parameter	Value	Unit
CPU	77.75	%
Throughput	1899	spectograms/s
Throughput	0.527	ms/spectogram

V. CONCLUSIONS

As shown in the confusion matrix in Figure 9, the command 'go' is the one with the lowest precision (79.6%) and 'no' has the lowest recall. This results show that both commands are being confused by the model, this can be solved by adding more real samples to both categories or augmenting the dataset. Also a manual review of the dataset might be needed, it was found that sample 'go/1bc45db9_nohash_1.wav' does not actually contain a valid sample. When presenting the model with other samples recorded by the us, all of the commands were accurately predicted with values close to 100% accuracy. As per results shown in Table IV each spectogram takes 0.527 to compute using the CPU which indicates the possibility of having realtime processing, however power consumption could be high considering the usage of the CPU was around 77.75%, this can be reduced by using an accelerator like a GPU.

REFERENCES

[1] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, "An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition," *Bell System Technical Journal*, vol. 62, no. 4, pp. 1035–1074, 1983. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1983.tb03114.x

[2] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," 2013. [Online]. Available: https://arxiv.org/abs/1303.5778

[3] A. Maas, Q. V. Le, T. M. O'Neil, O. Vinyals, P. Nguyen, and A. Y. Ng, "Recurrent neural networks for noise reduction in robust asr," in *INTERSPEECH*, 2012.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[5] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1764–1772. [Online]. Available: https://proceedings.mlr.press/v32/graves14.html

[6] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, 2nd ed. Prentice-hall Englewood Cliffs, 1999.

[7] J. G. Proakis and D. K. Manolakis, *Digital Signal Processing (4th Edition)*, 4th ed. Prentice Hall, 2006.

[8] E. Sejdić, I. Djurović, and J. Jiang, "Time–frequency feature representation using energy concentration: An overview of recent advances," *Digital Signal Processing*, vol. 19, no. 1, pp. 153–183, 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S105120040800002X

[9] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, 1937. [Online]. Available: https://doi.org/10.1121/1.1915893

[10] H. M. Fayek, "Speech processing for machine learning: Filter banks, mel-frequency cepstral coefficients (mfccs) and what's in-between," 2016. [Online]. Available: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

[11] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, p. e00938, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405844018332067

[12] W. S. Mcculloch and W. Pitts, "A Logical Calculus Of The Ideas Immanent In Nervous Activity," *Bulletin of Mathematical Biology*, vol. 52, no. l, pp. 99–115, 1990.

[13] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain." US, pp. 386–408, 1958.

[14] M. Minsky and S. Papert, "Perceptrons - an introduction to computational geometry," 1969.

[15] D. HUBEL and T. WIESEL, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," pp. 106–154, 1962.

[16] K. Fukushima and S. Miyake, "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position," *Pattern Recognition*, vol. 15, no. 6, pp. 455–469, 1982. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0031320382900243

[17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[18] R. V. Cox, S. F. De Campos Neto, C. Lamblin, and M. H. Sherif, "Itu-t coders for wideband, superwideband, and fullband speech communication [series editorial]," *IEEE Communications Magazine*, vol. 47, no. 10, pp. 106–109, 2009.

[19] B. McFee, A. Metsai, M. McVicar, S. Balke, C. Thomé, C. Raffel, F. Zalkow, A. Malek, Dana, K. Lee, O. Nieto, D. Ellis, J. Mason, E. Battenberg, S. Seyfarth, R. Yamamoto, viktorandreevichmorozov, K. Choi, J. Moore, R. Bittner, S. Hidaka, Z. Wei, nullmightybofo, A. Weiss, D. Hereñú, F.-R. Stöter, L. Nickel, P. Friesch, M. Vollrath, and T. Kim, "librosa/librosa: 0.9.2," Jun. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6759664