INTRODUCING MACHINE LEARNING IN UNDERGRADUATE DSP CLASSES

Uday S. Shanthamallu, Sunil Rao, Abhinav Dixit, Vivek S. Narayanaswamy, Jie Fan, Andreas Spanias
SenSIP Center, School of ECEE, Arizona State University
Email: {ushantha, rsunilsr, adixit8, vnaray29, jfan28, spanias}@asu.edu

ABSTRACT

Machine Learning (ML) and Artificial Intelligence (AI) algorithms are enabling several modern smart products and devices. Furthermore, several initiatives such as smart cities and autonomous vehicles utilize AI and ML computational engines. The current and emerging applications and the growing industrial interest in AI necessitate introducing ML algorithms at the undergraduate level. In this paper, we describe a series of activities to introduce ML in undergraduate digital signal processing (DSP) classes. These activities include a computational comparative study of ML algorithms for spoken digit recognition using spectral representations of speech. We choose spectral representations and features for speech as those concepts associate with the core topics in DSP such as FFT and autoregressive spectra. Our primary objective is to introduce undergraduate DSP students to feature extraction and classification using appropriate signal analysis and ML tools. An online module on ML along with a computer exercise are developed and assigned as a semester project in the DSP class. The exercise is developed in Python and also on the online JDSP HTML5 environments. An assessment study of the modules and computer exercises are also part of this effort.

Index Terms— Machine Learning, AI, Python, DSP education, Spectrogram, Online laboratory, J-DSP.

1. INTRODUCTION

Machine learning and artificial intelligence applications have grown rapidly across several disciplines, industries and cultures. Although some computer science and engineering curricula [1-3] include AI training, most Electrical Engineering undergraduate plans-of-study do not cover ML. On the other hand, many of the text books and survey papers have targeted graduate student audiences [4-6]. Several authors have argued that AI education must start early [1,7]. In this paper, we describe and assess an effort to introduce ML in our undergraduate Digital Signal Processing (DSP) classes. This activity includes developing online teaching modules and appropriate software to support computer exercises and projects. We started this activity initially for use in our SenSIP REU site program [8] and we developed a small J-DSP module and a k-means based exercise for our class and REU summer students [9]. At Arizona State, we teach core DSP at the senior level, though we have a DSP course for non-majors [10] as well. The DSP class is offered both as face-to-face and in a flipped course format [11-12]. We also routinely use some DSP materials for outreach and teacher training [13] using approaches such as those described in [14,20]. Introducing machine learning for all these activities is accomplished using a scalable approach and appropriate hands-on computer exercises. The module for the early DSP class has qualitative content and is accompanied by user-friendly J-DSP object oriented (block diagram based) software. The module for the senior level DSP class include Python and MATLAB programming [15] and therefore has a skill building component.

In the modules, we focus on learning from given data and then on producing meaningful results. In this particular effort, we expose students to feature extraction, training and clustering. We highlight problems and applications with big data sets such as those acquired from sensors and IoT [16,26] platforms. We also emphasize the need to understand patterns in order to avoid failures [17]. The activities are summarized as follows:

- We develop an online module that covers pre-processing, feature extraction, training and classification, along with a comparative study of ML algorithms. The module has a qualitative lecture for the early DSP class and a more rigorous lecture for the senior level DSP class.
- We develop functions and content to explain how ML can be used in spoken digit recognition. We associate feature extraction with topics taught in the DSP class.
- We develop software functions in Python and a computer exercise for the senior class. A J-DSP exercise is developed for the early level class for non majors.
- We assess all the content, software and exercises.

The rest of the paper is organized as follows. In Sections 2 and 3, we provide overviews of the module and exercise. Section 4 describes feature extraction aspects of the module and Section 5 provides background on different ML algorithms and Section 6 describes the Python functions. In Section 7, we demonstrate JDSP functions supporting the module. Section 8 discusses the assessment of the modules and Section 9 gives our concluding remarks.

2. THE MACHINE LEARNING MODULE

Our ML module for the senior DSP class consists of five online lectures that students can access through Blackboard. Each video-streamed lecture is assessed by a blackboard quiz that students complete. The quiz has two purposes: (i) to ensure that students go through the ML material and (ii) to assess the knowledge gained. The modules cover the conceptual diagram shown in Fig. 1.

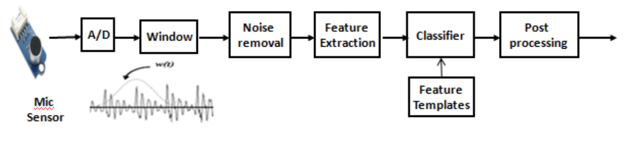

Figure 1: Typical signal processing framework from the sensor to the classifier.

The module begins with the pre-processing stage and followed by segmentation, de-noising and data interpretation by an appropriate ML algorithm. DSP operations include noise removal, feature extraction, and classification. The pre-processing operations have content associated with topics that are covered in the DSP class, such as the sampling theorem and quantization effects, FFTs and FFT-based de-noising and filtering, windows and segmentation. Feature extraction coverage includes descriptions of principal components and their compressive properties, FFT and linear prediction-based features, and cepstral parameters. In addition, the module covers FFT-based spectrograms that are common for "fingerprinting" audio in applications such as Shazam [18]. The connection of ML module and DSP class content is essential so that students can build on their knowledge. Again, the module has a qualitative version for the early DSP class for non-majors and a more rigorous version for the senior level class.

3. OVERVIEW OF THE COMPUTER EXERCISE

The computer exercise is built around a simple voice recognition task. This is motivated by the fact that many of the smart IoT products such as Google Home, Apple's Siri, Amazon's Echo [22,23] are enabled by speech recognition engines. More specifically, we focus on digit classification. Initially, we used formants derived from a linear prediction function and an object-oriented program to cluster formants and recognize digits. For the more advanced version, we use spectrograms to "fingerprint" digits and then use an appropriate machine learning algorithm to form clusters and classify the data. Details on spectrogram generation and its usefulness are provided in Section 4.

A general overview of our computer exercise is provided in Figure 2. We begin with spoken utterance of digits '0' through '4' which is available in a .wav file. We read the files and compute the spectrogram to obtain a digit spectral representation. For simplicity, we convert the spectrogram to a grayscale image resized to 28x28 pixels per image. We use the flattened version of this image (flattened from 28x28 to obtain 784 features) as an input to the three types of ML algorithms, namely: Simple logistic regression, multi-class Support Vector Machine (SVM) and an Artificial Neural Network (or Multilayer perceptron-MLP) with hidden layers. Each of the ML models is trained using training set and each model's performance is assessed using a test set. For training, we make use of 2,000 samples for each digit, total of 10,000 samples and testing set has 200 samples for each class. Details of different settings for various ML algorithms is provided in Section 5.

4. SPEECH SPECTROGRAM AND FEATURE EXTRACTION

Speech signals are non-stationary [21,27,29], and the spectral characteristics vary over time. Spectral characteristics are computed using the FFT on a frame-by-frame basis. The spectrogram is a two-dimensional plot of the frequency information over time with the spectral magnitude represented in color or grayscale. Distinctive aspects of the spectrograms for different digits enable digit classification. In Figure 3, we show spectrograms for spoken digits '0','1', and '2'.

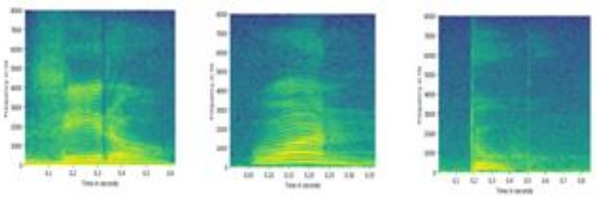

Figure 3: Spectrograms for spoken digits (left to right) '0', '1', and '2'. For each sub figure, x-axis represents time (in seconds) and y-axis represents frequency (in Hz).

5. MACHINE LEARNING MODULE

In this Section, we summarize three classification algorithms that we cover in the ML module, namely, Logistic Regression, Support Vector Machine (SVM) and Multi-layer perceptron (MLP) [28]. We represent the resized spectrogram image as $\tilde{\boldsymbol{x}} \in \mathbb{R}^{28\times28}$ and flatten it to obtain a vector $\boldsymbol{x} \in \mathbb{R}^{784}$ and the training set is represented by the matrix $\boldsymbol{X}_{train}$ which is of size $(10{,}000 \times 784)$. The testing set is represented by the matrix $\boldsymbol{X}_{test}$ of size $(1{,}000 \times 784)$. Since this is a supervised classification problem, each image $\tilde{\boldsymbol{x}}$ has a true label $\tilde{y} \in \{0,1,2,3,4\}$. Additionally, we convert the true label $\tilde{y}$ to $\boldsymbol{y} \in \mathbb{R}^5$, a 1-hot-encoded format as it is required for logistic regression and MLP.

(i) Logistic Regression

Multi-class logistic regression is an extension of simple logistic regression. Multinomial logistic regression is alternately referred as softmax classifier. For a softmax classifier, we first compute the score which is obtained through the linear transformation of the input features space.

$$\boldsymbol{s}^{(i)} = \boldsymbol{W} \cdot \boldsymbol{x}^{(i)} \qquad (1)$$

$\boldsymbol{s}^{(i)}$ represents the scores for the ith sample, $\boldsymbol{s} \in \mathbb{R}^5$ and it is unnormalized log probabilities for each class. $\boldsymbol{W}$ represents the weight or the model parameter for softmax classifier that is learned through backpropagation. The size of $\boldsymbol{W}$ is (5×784). We normalize the scores to obtain probabilities for each class.

$$P(\tilde{y} = k \mid \boldsymbol{s}_k) = \phi_{softmax}(\boldsymbol{s}_k) = \frac{e^{s_k}}{\sum_{j=0}^{4} e^{s_j}} \qquad (2)$$

$$\hat{y}^{(i)} = argmax(\phi_{softmax}(\boldsymbol{s}_k{}^{(i)})) \qquad (3)$$

$\hat{y}^{(i)}$ represents the predicted label for the given sample $\boldsymbol{x}^{(i)}$.

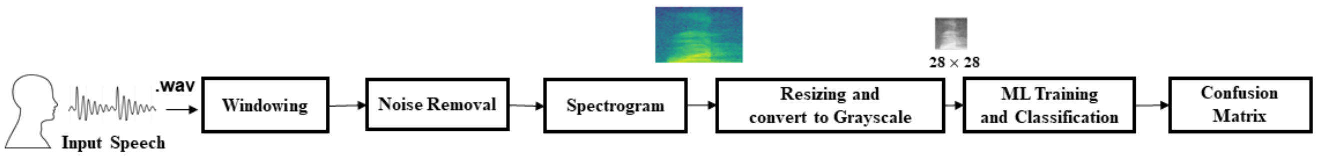

Figure 2: A block diagram showing the ML framework for spoken digit classification.

(ii) Support Vector Machine (SVM)

Support Vector Machines are supervised learning models that are used in classification and regression analysis. An SVM uses a hyperplane for classification and regression as well as outlier detection. For a binary classification, the general equation of the hyperplane is given as

$$\boldsymbol{x} \cdot \boldsymbol{w} + b = 0 \qquad (4)$$

Geometrically $\boldsymbol{w}$ represents the orientation of the hyperplane, b represents the bias and also gives the offset of the hyperplane from the origin. Once the SVM model is trained and the hyperplane is learned, label prediction follows the simple equation below:

$$\hat{y} = +1 \; if \; \boldsymbol{x} \cdot \boldsymbol{w} + b > 0$$
$$\hat{y} = -1 \; if \; \boldsymbol{x} \cdot \boldsymbol{w} + b < 0$$

We extend binary SVM to the multi-class classification problem, specifically, we choose one-vs-all classifiers and thus train 5 SVM models to do a 5-class classification.

(iii) Multi-layer Perceptron

Neural networks are constructed by stacking layers of neurons, allowing it to learn multiple representations of the input data. The outputs of each neuron in each layer is fed as a weighted input to the next layer. The final layer is the output layer which is a simple softmax classifier. Information flows through the neural networks in two ways: (i) In *forward propagation* the MLP model predicts the output for the given data and (ii) In *backpropagation* the model adjusts its parameters considering the error in the prediction. The *activation* function used in each neuron allows the MLP to learn a complex function mapping. The MLP architecture used for spoken digit classification is shown in Figure 4. Input to the model is flattened spectrogram $\boldsymbol{x}$, the output of the first and second hidden layer is given by

$$\boldsymbol{h}_1 = \sigma(\boldsymbol{W}_1 \cdot \boldsymbol{x} + \boldsymbol{b}_1) \qquad (5)$$
$$\boldsymbol{h}_2 = \sigma(\boldsymbol{W}_2 \cdot \boldsymbol{h}_1 + \boldsymbol{b}_2) \qquad (6)$$

Where $\boldsymbol{W}_1$of size (16×784) is weight matrix from input to first hidden layer and $\boldsymbol{W}_2$ of size (8×16) is the weight matrix from first hidden layer to second hidden layer. $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are biases of size 16 and 8 respectively. Finally, the output of the MLP is obtained as:

$$\hat{y} = argmax(\phi_{softmax}\,(\boldsymbol{h}_2))$$

Note that this resembles the softmax equation from multinomial logistic regression.

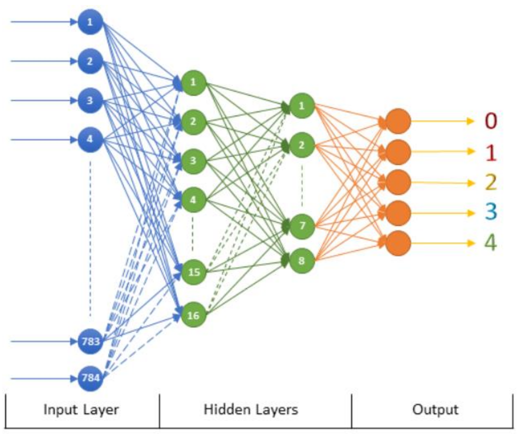

Figure 4: Artificial Neural Network with two hidden layers with 16 and 8 neurons respectively. Input to the neural network is the resized spectrogram image and output of the neural network is the probability for each class.

6. PYTHON FUNCTIONS

For the senior level exercise, we implemented and provided the three ML algorithms as functions in Python [24]. We also provided MATLAB support for those functions which may also have real-time DSP extensions [25]. Students are given these functions and asked to write the main program in order to perform the training and classification for the spoken digit from the speech recordings. Before completing this task, students are given a lecture explaining the functions in Python and MATLAB and are provided sample code for simple classification tasks. Students are then asked in the computer exercise to compare the algorithms and provide results in a confusion matrix plot as shown in Figure 5.

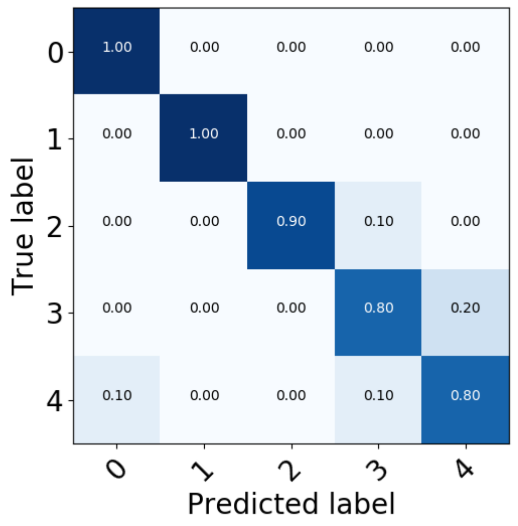

Figure 5: Confusion matrix for Spoken Digit Classification.

The pedagogical value goes beyond machine learning as it exposes students to Python programming which has become very common in ML tasks. Students are given sufficient assistance to complete the hands-on tasks which they describe in a report that they submit. Students are asked to complete a quiz to further assess their understanding of all tasks. In the assessment stage, students are also interviewed to provide comments on their overall learning experience.

7. J-DSP ONLINE FUNCTIONS

For the DSP class for non-majors, we develop the same exercise in the online block-diagram type software JDSP-HTML5. A special ML qualitative module is video streamed to prepare them for the exercise. The J-DSP software (now in HTML 5) can be used to perform signal processing and ML experiments. The user interface of the software provides options to change the parameters of the ML algorithm. The block diagram for training and classification using the SVM algorithm is shown in Fig 6. J-DSP embeds ML algorithm blocks which users can employ to for training and classification.

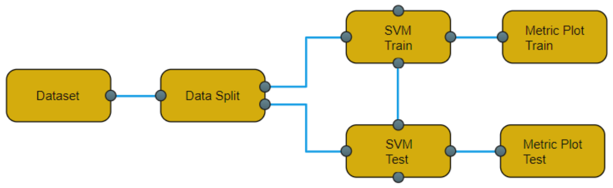

Figure 6: Block diagram in JDSP-HTML5 showing data splitting, SVM training and testing. The *Metric Plot* block provides the confusion matrix after classification.

The *Dataset block* contains several combinations of data that can be used to perform machine learning experiments. The data from *Dataset block* is sent to *Data Split* block where the user can split the data in two fractions of randomized data. For example, the data can be separated into 75% for training and 25% for testing. Data splitting after randomizing is a common practice in machine learning. The first output of the *Data split* block is sent to the *SVM block* where an SVM model can be trained. The trained SVM model from the *SVM train block* and the second output of the *Data Split block is* sent to *SVM Test block* where it can be used for testing the performance of the trained SVM model. The output of the SVM train model is used in *Metric Plot* block to visualize the decision boundary and plot confusion matrix. The *Metric Plot* block shows the region of the classified classes and how well the model has fit the provided training data. Similarly, the output of SVM Test data can be used to plot the confusion matrix on the testing data. The confusion matrix on the testing data shows the robustness and accuracy of the trained model and demonstrates how well it performs on unseen data. It is important for the students to see validation and cross validation experiments with the different algorithms.

8. ASSESSMENT

The assessment of the modules and exercises is based on pre- and post-quizzes and interviews. Feedback forms are initially collected from students for general assessment of the quality of educational materials, the software quality and accessibility. The students then access the learning software and specific instructions pertaining to the exercise. Students are asked to compare algorithms for performance and complexity. Metrics for performance include exercise related questions on spectrograms, extraction of features from spectrograms and general understanding of Python. Additional assessments based on pre-quiz and post-quiz of the exercise aim to determine the student's learning of the specified topics. Some preliminary assessment results have already been compiled for the J-DSP exercise during Spring 2018 [15] and Fall 2018. The pre- and post-quiz assessment (Fig. 7) is based on questions listed in [19] which addressed the knowledge gained on: speech frame size and statistics {q.1), the k-means training and clustering process (q.2. and q.3), and the speech feature selection (q.4. and q.5),. The average improvement is around 50%. More results on formative and summative assessment will be compiled and presented in the final paper.

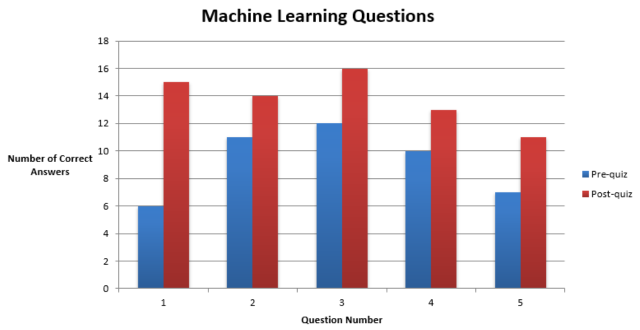

Figure 7: Pre-assignment and post-assignment quiz result. On the x-axis we have the question numbers, and on y-axis we have the number of students who answered the question correctly. Details on the posted questions are given in [19].

9. CONCLUSIONS

In this paper, we described a module and computer exercise to introduce Electrical Engineering students taking the DSP class to machine learning. Two modules have been developed to be used in a sophomore DSP class for non-majors and a senior level class for EE majors respectively. The ML content for the sophomore class for non-majors is qualitative and the computer exercise is based on a user-friendly J-DSP block diagram-based software. For the senior DSP class the module is more rigorous and includes Python and MATLAB programming and comparisons of three different algorithms. The algorithmic task is to recognize digits from spectrogram "fingerprints" using machine learning. The students reported on performance and complexity and responded to quizzes to assess the knowledge gained. Complete assessment results and details on the quizzes, assignments, Python and MATLAB software evaluation, and overall assessment process will be given in the final paper.

ACKNOWLEGEMENTS

The work at Arizona State University is supported in part by the NSF DUE award 1525716 and the SenSIP Center.

REFERENCES

[1] M. Kandlhofer, G. Steinbauer, S. Hirschmugl-Gaisch, P. Huber, “Artificial intelligence and computer science in education: From kindergarten to university,” 2016 IEEE Frontiers in Education Conference (FIE) (2016), Eire, PA.

[2] L. Torrey, "Teaching Problem-Solving in Algorithms and AI", 3rd Symposium on Educational Advances in Artificial Intelligence, 2012.

[3] T. Barik, M. Everetty, R. E. Cardona-Rivera, D. L. Roberts, E. F. Gehringer, "A Community College Blended Learning Classroom Experience through Artificial Intelligence in Games", IEEE Frontiers in Education Conference (FIE), pp. 1525-1531, 2013.

[4] Danilo Mandic and Jonathon A. Chambers, Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures, Wiley 2001.

[5] S. Theodoridis, Machine Learning: A Bayesian and Optimization Perspectives, Academic Press, April 2, 2015.

[6] U. S. Shanthamallu, A. Spanias, C. Tepedelenlioglu and M. Stanley, "A brief survey of machine learning methods and their sensor and IoT applications," *2017 8th International Conference on Information, Intelligence, Systems & Applications (IISA)*, Larnaca, August 2017.

[7] Mathews, V. (2018). How teaching AI in schools could help equip students for future careers. [online] Raconteur. Available at: https://www.raconteur.net/technology/ai-in-schools-students-future [Accessed 23 Oct. 2018].

[8] A. Spanias and J. Blain Christen, “A STEM REU Site on The Integrated Design of Sensor Devices and Signal Processing Algorithms',” *Proc. IEEE ICASSP 2018,* Calgary, April 2018.

[9] A. Dixit, U. Shanthamallu, A. Spanias, S. Rao, S. Katoch, M. Banavar, G. Muniraju, J. Fan, P. Spanias, A. Strom, C. Pattichis, H. Song, "Multidisciplinary Modules on Sensors and Machine Learning," " *Proc. 2018 ASEE Annual Conference*, Salt Lake City, June 2018.

[10] A. Spanias, "An introductory signal processing course offered across the curriculum, “IEEE *Signal Processing and Signal Processing Education Workshop (SP/SPE)*, Utah, August 2015.

[11] W. U. Bajwa, "On flipping a large signal processing class", IEEE Signal Processing Mag., vol. 34, no. 4, pp. 158-170, Jul. 2017.

[12] B. Van Veen, "Flipping signal-processing instruction", IEEE Signal Processing Mag., vol. 30, no. 6, pp. 145-150, Nov. 2013.

[13] SenSIP Outreach, https://sensip.engineering.asu.edu/outreach/

[14] M. F. Bugallo and A. M. Kelly, "Engineering outreach: Yesterday, today and tomorrow," *Signal Processing Magazine*, vol. 34, pp. 69-100, May 2017.

[15] A. Dixit, U. S. Shanthamallu, A. Spanias, V. Berisha, and M. Banavar, "Online Machine Learning Experiments in HTML5"*, IEEE Frontiers In Education (FIE)*, San Jose, October 3-6, 2018.

[16] M. Stanley and Jong Ming Lee, Sensors for IoT Applications, ISBN 9781627054638, Synthesis Lectures, Morgan and Claypool Publishers, 113 Pages, March 2018,

[17] Wakabayashi, Daisuke, "Self-Driving Uber Car Kills Pedestrian in Arizona, Where Robots Roam." The New York Times, 2018.

[18] Avery Li-Chun Wang, "An Industrial-Strength Audio Search Algorithm," *ISMIR 2003*, Oct. 2003.

[19] Diagnostic Quiz, Linear Prediction and Machine Learning, Website: http://jdsp.engineering.asu.edu/MLExercise/MLQuiz.pdf

[20] H. Godrich, A. Nehorai, A. Tajer, M. Sabrina Greco, C. Zhang, Special Article Series on Signal Processing Education via Hands-On and Design Projects, From the Guest Editors. *IEEE Signal Process. Mag.* 34(1): 13-15, 2017.

[21] Venkatraman Atti. Algorithms and Software for Predictive and Perceptual Modeling of Speech, Lecture Series, Morgan and Claypool Publishers, Ed. A. Spanias, ISBN: 9781608453870, March 2011.

[22] Sally Applin, "Amazon's Echo Look: Harnessing the Power of Machine Learning or Subtle Exploitation of Human Vulnerability?", IEEE Consumer Electronics Magazine, Vol. 6, Issue: 4, Oct. 2017) pp. 125 - 127, Oct. 2017.

[23] Paul Dempsey, "The teardown: Google Home personal assistant," IET Engineering & Technology, Volume: 12, Issue: 3 , April 2017.

[24] Maurice Charbit, Digital Signal Processing (DSP) with Python Programming, Wiley, 2017.

[25] Thad B. Welch, Cameron H.G. Wright, Michael G. Morrow, Real-Time Digital Signal Processing from MATLAB to C with the TMS320C6x DSPs, CRC Press, 2017.

[26] B. Mears and M. Shah, Virtual Design of an Audio Lifeloggin System: Tools for IoT Systems, Synthesis Lectures on Algorithms and Software in Engineering, Morgan & Claypool Publishers, Ed. A. Spanias, ISBN:9781627056717, 2016.

[27] A. Spanias, T.Painter, V.Atti, Audio Signal Processing and Coding, Wiley Publishers, March 2017.

[28] H. Song, J.Thiagarajan, P.Sattigeri, A.Spanias, “Optimizing Kernel Machine using Deep Learning”, IEEE Trans. On Neural Networks and Learning Systems, NLS-2017-p-8053.R1, pp. 5528-5540, Feb. 2018.

[29] A. Spanias, “Speech Coding”: A Tutorial Review”, Proceedings of IEEE, Vol.82, No. 10, pp. 1441-1582, October 1994.