9th International Conference on Speech Prosody 2018
13-16 June 2018, Poznań, Poland

Sparse Coding of Pitch Contours with Deep Auto-Encoders

Nicolas Obin[1], *Julie Belião*[2]

[1] IRCAM, CNRS, Sorbonne Université, Paris, France
[2] Unbabel, Lisbon, Portugal

Abstract

This paper presents a sparse coding algorithm based on deep auto-encoders for the stylization and the clustering of pitch contours. The main objective of the proposed algorithm is to learn a set of pitch templates that can be easily interpreted by humans and whose combination can approximate efficiently the observed pitch contours. The proposed learning architecture is based on deep auto-encoders, commonly used to learn non-linear and low-dimensional latent representations that approximate the observed data. The proposed deep architecture is based on stacked auto-encoders and the sparsity of the network is investigated in order to learn a more robust and general representation of the pitch contours (dropout, denoising auto-encoder, sparsity regularization). The deep auto-encoding of the pitch contours is illustrated and discussed on the TIMIT American-English speech database[†] with comparison of other existing stylization and clustering algorithms.

Index Terms: speech prosody, pitch contour, sparse coding, deep auto-encoders

1. Introduction

The representation of pitch contours is a fundamental task in the modeling of speech prosody, from the construction of linguistic models for the understanding of speech prosody and its interaction with other linguistic domains to the learning of generative models for text-to-speech synthesis applications [1]. The computation of such representations can be decomposed into: the *estimation* of the fundamental frequency (F_0) (alternatively, "pitch") of the speech signal and its *segmentation* into speech units, the *stylization* of the pitch variations by removing their undesired variability, and its *encoding* into meaningful labels [2, 3] or templates [4, 5, 6] that can be interpreted by humans and learned by machines.

On the one side, stylization of pitch contours is generally achieved by preserving the desired variability of the pitch variations, generally described as their slow time variations. This can be simply done by low-pass filtering the pitch variations [2] but more generally by decomposing the pitch variations onto a set of previously defined bases with some desired properties: typically smoothed or slowly time-varying bases such as polynomial [7, 8, 9], cosinusoidal [10], and B-splines [11]. Mathematically speaking, this decomposition is equivalent to a dimensionality reduction of the observed pitch contours since their are represented into the space defined by the bases, whose number is smaller than the number of values of the pitch contours. Besides, this decomposition may in turn be interpreted as pitch templates from which any observed pitch contours can be reconstructed by linear combination of the bases. The obtained stylization can serve as a "denoising" prior to some advanced encoding of the pitch contours [2] or can be used directly as encoding [12, 13]. On the other side, clustering has been used for the prototyping and the visualization of pitch contours. State-of-the-art pitch contour clustering algorithms include: k-means and weighted k-means [14, 15], self-organizing maps (SOM, [16, 17]), functional principal component analysis (FPCA, [18]), and recurrent neural networks (RNN, [5]). Clustering can be seen as a dimensionality reduction technique, in which the bases used for decomposition are no longer fixed a priori but learned from the data. The reduction is achieved by assigning each pitch contour to a cluster, and reducing its variability to the centroid of this cluster. The centroids are representative of the cluster and assumed to encode the meaningful pitch information, and will further be referred to as *pitch templates*. One important assumption about these clustering techniques is that it is usually a *hard* clustering: each data is associated to one-and-only-one centroid (k-means, SOM), while the observed pitch contours may actually be the result of a combination of some latent and underlying pitch templates.

This paper establishes a sparse coding algorithm based on deep neural network (DNN) for the modeling and the representation of pitch contours. The main originality of the proposed approach is to learn the bases used for the decomposition of pitch contours from the speech database, by assuming that the observed pitch contours result from a (linear or non-linear) combination of the underlying learned pitch templates. To do so, the learning of the pitch templates is achieved by deep auto-encoders, whose principle is to determine a latent representation of the pitch contours in a low-dimensionality space which best approximates the observed pitch contour. A deep encoder is a deep neural network used for dimensionality reduction to learn a latent representation of the data in a low dimension space which best approximates the data. The deep auto-encoder allows to learn non-linear relationships between the observed data and the latent representation of the hidden layers. Deep auto-encoders can be learned by pre-training using restricted Boltzman machines (RBM-AE [19]) or stacked auto-encoders (SAE, [20]). Besides, sparsity of the network is generally encouraged [21], specializing subparts of the network to some specific aspects of the data. Sparse networks provides a better interpretation of the network, compression of the data, and may also be a way to prevent over-fitting and increase robustness to noise. Sparsity can be obtained by sparsity regularization [22], dropout [23], and denoising auto-encoders [24]. In this paper, sparse auto-encoders will be used in order to improve the interpretability of the pitch templates learned by the network.

†. The resource produced in this paper is freely available on : `https://github.com/nicolasobin/`.

The remaining of the paper is organized as follows: in Section 2 we describe the deep auto-encoder architectures used to encode, with a focus on the sparsity of the network. The proposed algorithms are illustrated and discussed in Section 3 on a task of pitch contour stylization and clustering from the TIMIT database [25].

2. Deep Coding of Pitch Contours

The proposed algorithm for the coding of pitch contours is based on deep auto-encoders, a family of unsupervised deep neural networks commonly used for the learning of latent representation of data in some low-dimensional space which concentrates the meaningful information used to encode the data. This section presents the fundamentals of a deep auto-encoder and some of its variants, including sparsity regularization and denoising auto-encoder.

2.1. Auto-encoder

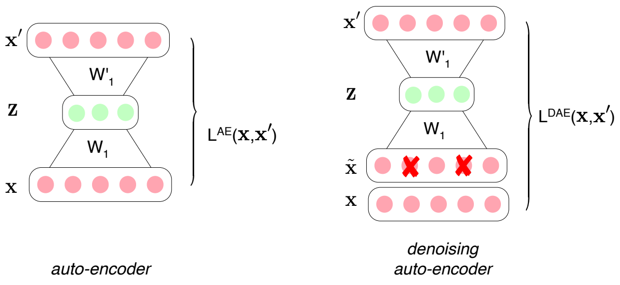

FIGURE 1 – Illustration of an auto-encoder and a denoising auto-encoder with masking noise.

The auto-encoder (AE) is a single-layer feed-forward neural network composed of an input layer, an hidden layer, and an output layer with the same number of neurons that the input layer. The particularity of the auto-encoder is the neural network is trained so as to reconstruct its original inputs. An auto-encoder can be decomposed into an encoder and a decoder. The encoder is a deterministic function $\mathbf{f}$ that transforms an input vector $\mathbf{x}$ into a latent representation vector $\mathbf{z}$, and defined as a affine function followed by a non-linear function s

$$\mathbf{z} = f(\mathbf{x}) = s(\mathbf{W}\mathbf{x} + \mathbf{b}) \quad (1)$$

Then, the decoder transforms the latent vector $\mathbf{z}$ into the reconstructed vector $\mathbf{x}'$

$$\mathbf{x}' = f'(\mathbf{z}) = s(\mathbf{W}'\mathbf{z} + \mathbf{b}') \quad (2)$$

The loss function to be minimized by the AE can be written as the quadratic error between the input vector $\mathbf{x}$ and the reconstructed vector $\mathbf{x}'$

$$L(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'||^2 \quad (3)$$

where $||.||$ denotes the L-2 norm.

A sparse network can be encouraged by adding regularization terms on the parameters of the neural network

$$L(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'||^2 + \alpha||\mathbf{W}||^2 + \beta\ \rho(f(\mathbf{x})) \quad (4)$$

where $||\mathbf{W}||^2$ is the square L-2 norm of the network weights, ρ the desired average activation of the network, α and β the coefficient associated with the weight an activation regularizations.

The parameters of the auto-encoder $\theta = \{\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{b}'\}$ are determined by minimization of the loss function:

$$(\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{b}') = \underset{\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{b}'}{\operatorname{argmin}}\ L(\mathbf{x}, \mathbf{x}') \quad (5)$$

This unconstrained optimization problem is solved using back-propagation [26] by stochastic gradient descent (SGD).

2.2. Denoising Auto-encoder

A denoising auto-encoder is a simple variant of the auto-encoder in which the input $\mathbf{x}$ is replaced by $\tilde{\mathbf{x}}$, a corrupted version of $\mathbf{x}$ [24], and the corresponding reconstructed output vector $\tilde{\mathbf{x}}'$. The main difference to the auto-encoder is that the loss function is not written as a function of the corrupted input vector $\tilde{\mathbf{x}}$, but to its uncorrupted version $\mathbf{x}$ so that

$$L(\mathbf{x}, \tilde{\mathbf{x}}') = ||\mathbf{x} - \tilde{\mathbf{x}}'||^2 \quad (6)$$

Thus, the idea of a denoising auto-encoder is to determine the best approximation of the uncorrupted data from corrupted data, i.e. *denoising* the corrupted data. In other words, the denoising auto-encoder learns a latent representation of the data which is insensitive to noise. Various types of corruption can be used, among them additive Gaussian noise and masking noise are among the most popular. An illustration of the auto-encoder and its denoising auto-encoder variant is presented in Figure 1.

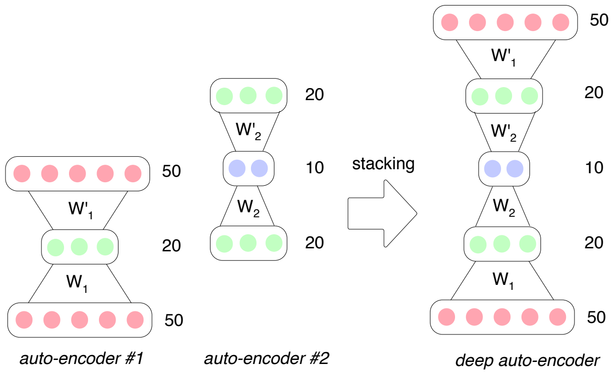

FIGURE 2 – Illustration of a stacked auto-encoder. On the left: auto-encoders as used for pre-training ; on the right: the resulting deep auto-encoder obtained by stacking the auto-encoders as used for fine-tuning.

2.3. Deep Auto-encoder

A deep auto-encoder is the generalization of the single-layer auto-encoder, by adding more hidden layers in order to form a deep architecture.

2.3.1. Pre-training

Due to the sensitivity of the learning of a deep auto-encoder to its initialization, generally random, pre-training algorithms have been proposed based on layer-wise learning algorithms, such a Restriced Bolzman Machines (RBM, [19]) and Stacked Auto-Encoder (SAE, [20]). A stacked auto-encoder is a symmetric deep auto-encoder, composed of the stacking of single-layer auto-encoders. In a stacked auto-encoder, the first layer is learned from the input data and the intermediate layers k ($k > 1$)

can be written with the following encoding, decoding, and loss functions

$$\mathbf{z}^k = f^k(\mathbf{z}^{k-1}) = s(\mathbf{W}^k\mathbf{y}^{k-1} + \mathbf{b}^{(k)}) \quad (7)$$

$$\mathbf{x}'^k = f'^k(\mathbf{z}^k) = s(\mathbf{W}'^k\mathbf{z}^k + \mathbf{b}'^k) \quad (8)$$

$$L^k(\mathbf{y}^{k-1}, \mathbf{x}'^k) = ||\mathbf{x}^k - \mathbf{x}'^k||^2 \quad (9)$$

The layers are trained one by one, starting by learning the first auto-encoder from the input observed data, and propagating the output latent representation as the input of the next auto-encoder, and so on. The pre-trained layers are then stacked to form the deep auto-encoder, as illustrated in Figure 2.

2.3.2. *Fine-tuning*

The pre-trained stacked auto-encoders are then used as an initialization for the fine-tuning of the entire deep network. The loss function is now defined as the error of the whole network. The gradient of the loss function with respect to the weights of the network is computed by applying iteratively the chain rule to each layer, from which the back-propagation is used to update the weights of the network.

3. Experiment

This section investigates the deep auto-encoding of pitch contours on syllables of English-American speakers, with comparison of other existing stylization and clustering algorithms.

3.1. Speech material and preprocessing

An experiment has been conducted on the TIMIT American-English speech database [25]. The TIMIT database is composed of 10 sentences read by 630 American-English speakers for a total of 6,300 speech utterances. The TIMIT database comes with phoneme and word alignment, but without syllable alignment. The reference syllable alignment has been obtained with the NIST syllabification software [27], and manually checked as previously described in [28], with a total of 77,391 syllables. The fundamental frequency of the speakers was estimated by using the SWIPE algorithm [29] with a minimum pitch value of 75 Hz, a maximum pitch value of 350 Hz, and a hop size of 5 ms, without any post-processing for correcting or smoothing the raw pitch values. The voiced/unvoiced decision was computed from the pitch strength associated with the pitch value estimate with a threshold of 0.25 (the pitch strength being a value between 0 and 1 corresponding to the periodicity of the speech frame).

Then, the following processing were applied to form the pitch contours database used for learning. First, a pitch contour was calculated for each syllable by identifying the longest voiced sequence over the syllable. The unvoiced syllables and the syllables for which the voicing was too short ($\leq$ 25 ms, i.e., 5 pitch values) were discarded. Then, the pitch contours were centered around 0, by subtracting their mean value. Finally, the pitch contours were time-normalized, by resampling the pitch contours of variable size to $N_{f0} = 50$ pitch values using a linear interpolation. This resulted into a matrix of 74,284 pitch contours with 50 values each.

3.2. Benchmark and Methodology

The following state-of-the-art clustering algorithms considered for comparison were: k-means on the raw pitch contours, a k-means on the 5-order DCT of the pitch contours [10], and a k-means on the 5-dimension PCA of a full dimension DCT (considered as a reasonable approximation of the FPCA, as described in [18]). The following variants of the deep auto-encoders were considered for comparison: a standard stacked auto-encoder (SAE), a stacked denoising auto-encoder (SDA), and a stacked denoising auto-encoder with sparsity regularization expressed as the square L-2 norm of the weights (SDA + sparsity). The following architectures were examined: from 1 to 3 hidden layers, each layer having a number of neurons in $[5, 10, 20]$. Sigmoid activation functions were used for all layers except for the last layer for which a hyperbolic tangent activation function was used to allow the network to output positive and negative values. The deep auto-encoders were learned with mini-batches of 100 pitch-contours, a learning rate of 1, a momentum of 0.5, and by using 50 epochs were used for pre-training and 500 epochs for fine-tuning. The fraction of masked neurons in the denoising auto-encoder was comprised in $[0, 0.1, 0.25, 0.5]$ and used for all input layers of the stacked auto-encoders. The sparsity constraint on the weights was comprised in $[0., 0.0001, 0.001]$, and only during pre-training.

The clustering algorithms were compared by mean of the reconstruction error and the clustering consistency. The reconstruction error is measured by using the root mean square error (RMSE, in Hz) between the pitch contour and its reconstruction, defined as

$$RMSE(\mathbf{x}) = \frac{1}{\sqrt{N_{f0}}}||\mathbf{x} - \mathbf{x}'|| \quad (10)$$

The clustering consistency is measured by using the silhouette coefficient [30] defined as

$$sc(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max(a(\mathbf{x}), b(\mathbf{x}))} \quad (11)$$

where $a(\mathbf{x})$ is the mean intra-cluster distance in which $\mathbf{x}$ belongs and $b(\mathbf{x})$ is the mean nearest-cluster distance from $\mathbf{x}$. The silhouette coefficient lies between -1 and $+1$, indicating respectively the limits of a bad and a good clustering of the data. The silhouette index (SI) is defined as the overall average silhouette coefficient. The clustering is considered as fairly consistent for silhouette values greater than 0.5.

The two measures are complementary to assess the quality of the modeling of the pitch contours: the RMSE indicates the accuracy of the reconstructed pitch contour compared to the original pitch contour, and the silhouette index indicates the consistency of the clusters created by the reconstructed pitch contours. Though both measures are important, the consistency of the clusters may be a better indicator of the coding efficiency: indeed, a good reconstruction may encode unnecessary details of the pitch contours while a good clustering means that the encoded pitch contours are organized into well-structured clusters. In other words, this means that the encoded pitch contours are concentrated around well-defined and distinctive pitch templates, suggesting that the meaningful information contained in the pitch contours has been encoded.

3.3. Results and Discussion

The RMSE and SI obtained for the benchmark algorithms is presented in Table 1. When the approximation and clustering algorithms are different, the approximation is reported in the RMSE and the clustering is reported in the silhouette. The

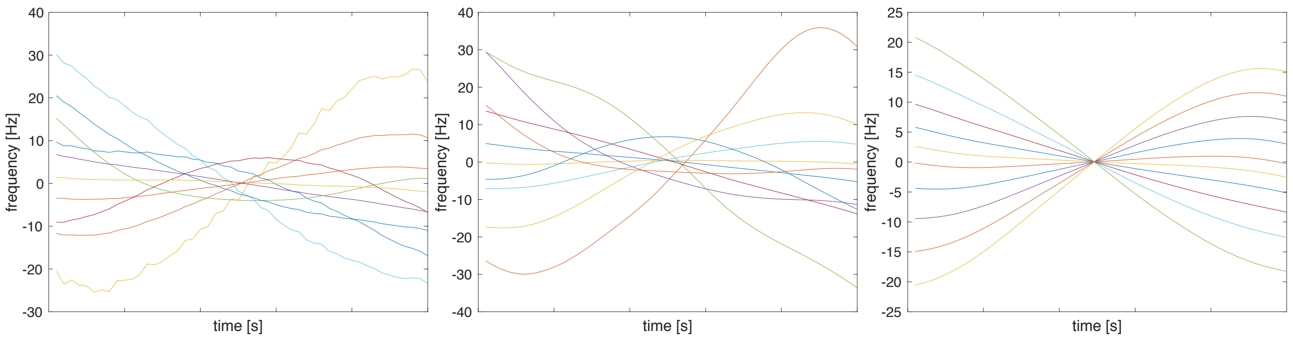

FIGURE 3 – Pitch contours bases obtained by k-means clustering in the low-dimensional representation of the auto-encoder. From left to right: stacked auto-encoder, stacked denoising auto-encoder, and stacked denoising auto-encoder with a strong sparsity constraint.

k-means algorithm has a poor reconstruction (2.72 Hz) and a fair clustering (40.9%), the poor approximation being easily explained by the hard clustering nature of the k-mean algorithm: the data is approximated by the centroid of its closest cluster only. Since the k-mean algorithm is by nature defined to minimize the RMSE with a single cluster, the SOM has a similar but slightly higher RMSE (2.74 Hz) but with a higher clustering consistency (41.8%). The 5-DCT approximation provides a close approximation of the pitch contours with only 5 coefficients (0.91 Hz), but which comes to a lower clustering consistency (37.4%). Finally, the 5-dimension PCA on the full-order DCT provides a trade-off between approximation (1.29 Hz) and clustering consistency (42.1%).

By comparison, the original SAE provides the best approximation (0.69 Hz) together with a fairly consistent clustering (40.0%). The SDA version provides a higher approximation but with a very substantial increase of the clustering consistency (48.4%). Finally, the addition of sparsity regularization has a similar approximation but with the highest clustering consistency (51.5%). The main finding is that the deep architecture of the SAE combines a low reconstruction error together with a high clustering consistency within a single framework. Also, the sparsity constraint naturally lead to a slight increase of the reconstruction error but with a very large increase of the clustering consistency (SDA and SDA + sparsity regularization vs. SAE).

This first tendency is confirmed by inspection of the pitch templates learned by the deep auto-encoders as illustrated in Figure 3. The pitch templates learned by the original SAE network appear to be consistent with the expected pitch clusters, but somehow noisy. The pitch templates learned by the SDA + sparsity network are naturally smoothed and consistent, the smoothness being the counterpart of the substantial gain in clustering consistency. The limit case of the sparsity regularization is also illustrated on the top right of the Figure, revealing a latent family of elementary functions (similar to some extent to the elementary bases of the DCT) that are used to construct more complex pitch templates.

The effect of sparsity on the reconstruction and the clustering is confirmed by multiple correlation analysis [31] between the RMSE and the silhouette value as dependent variables and the fraction of masked neurons and the weight sparsity as explanatory variables. The RMSE is significantly correlated with the masking fraction (correlation coefficient $c = 0.43$, corresponding value $p = 0.05$), not with the sparsity ($c = 0.29$, $p = 0.21$), and highly significantly correlated with the two factors together ($c = 0.51$, $p = 0.02$). The silhouette has similar correlation properties: $c = 0.43$ and $p = 0.05$ for the masking noise, $c = 0.29$ and $p = 0.21$ for the weight sparsity, and $c = 0.48$ and $p = 0.02$ for the two factors together. This clearly indicates that the masking noise is effective alone for the approximation and the clustering of the pitch contours, though the weight sparsity alone is not. Nevertheless, the weight sparsity becomes effective when combined with the masking noise.

	RMSE (Hz)	SI (× 100)
k-means	2.72	40.9
SOM	2.74	41.8
5-DCT + k-means	0.91	37.4
DCT + 5-PCA + k-means	1.29	42.1
SAE	0.69	40.0
SDA	1.55	48.4
SDA + sparsity	1.51	51.5

TABLE 1 – Reconstruction and clustering scores obtained by the considered algorithms.

4. Conclusion

This paper presented a sparse coding of pitch contours for the clustering and the visualization of pitch contours. The proposed algorithm is based on deep auto-encoders with sparsity constraints including denoising auto-encoders and sparsity regularization in order to encourage the network to learn meaningful latent representation of the pitch contours. It is examined for the coding of pitch contours of American-English speakers of the TIMIT database, and compared to other existing stylization and clustering algorithms. The proposed algorithms have the best compromises between the reconstruction error and the clustering consistency of the pitch contours encoded by the network, as compared to other existing stylization and clustering algorithms. Future research will integrate the deep coding of pitch contours for their automatic labeling and investigate the coding of pitch contours on more expressive speech, such as emotional speech. Finally, the proposed coding can be used for the modeling of speech prosody for speech synthesis and voice conversion.

5. References

[1] N. Obin, “MeLos: Analysis and Modelling of Speech Prosody and Speaking Style,” PhD. Thesis, IRCAM - UPMC, 2011.

[2] N.Obin, J. Beliao, C. Veaux, and A. Lacheret, “SLAM: Automatic Stylization and Labelling of Speech Melody,” in *Speech Prosody*, 2014, p. 246–250. [Online]. Available: https://github.com/jbeliao/SLAM/

[3] A. Rosenberg, “AutoBI-a tool for automatic toBI annotation,” in *Interspeech*, 2010, pp. 146–149.

[4] R. Dall and X. Gonzalvo, “JNDSLAM: A SLAM extension for speech synthesis,” in *Speech Prosody*, 2016, p. 1024–1028.

[5] S. Ronanki, G. E. Henter, Z. Wu, and S. King, “A template-based approach for speech synthesis intonation generation using LSTMs,” in *Interspeech*, 2016, p. 2463–2467.

[6] J. A. Louw, A. Moodley, and A. Govender, “The Speect text-to-speech entry for the Blizzard Challenge 2016,” in *Interspeech*, 2016.

[7] E. Grabe, G. Kochanski, and J. Coleman, “Quantitative modelling of intonational variation,” in *Speech Analysis and Recognition in Technology, Linguistics and Medicine*, 1994, pp. 1–23.

[8] P. Taylor, “Analysis and synthesis of intonation using the TILT model,” *Journal of the Acoustic Society of America*, vol. 107, pp. 1697–1714, 2000.

[9] T. Mishra, J. Van Santen, , and E. Klabbers, “Decomposition of Pitch Curves in the General Superpositional Intonation Model,” in *Speech Prosody*, Dresden, Germany, 2006.

[10] J. Teutenberg, C. Watson, and P. Riddle, “Modelling and Synthesising F0 contours with the Discrete Cosine Transform,” in *International Conference on Acoustics, Speech, and Signal Processing*, Las Vegas, U.S.A, 2008, pp. 3973–3976.

[11] D. Lolive, N. Barbot, and O. Boëffard, “Melodic contour estimation with B-spline models using a MDL criterion,” in *International Conference on Speech and Computer*, Saint Petersburg, Russia, 2006, pp. 333–338.

[12] N. Obin, A. Lacheret, and X. Rodet, “Stylization and Trajectory Modelling of Short and Long Term Speech Prosody Variations,” in *Interspeech*, Florence, Italy, 2011, pp. 2029–2032.

[13] X. Yin, M. Lei, Y. Qian, F. K. Soong, L. He, Z.-H. Ling, and L.-R. Dai, “Modeling F0 trajectories in hierarchically structured deep neural networks,” *Speech Communication*, vol. 76, pp. 82–92, 2016.

[14] E. Klabbers and J. P. H. van Santen, “Clustering of foot-based pitch contours in expressive speech,” in *ISCA Speech Synthesis Workshop*, 2004, pp. 73–78.

[15] O. Migliore and N. Obin, “At the interface of speech and music: A study of prosody and musical prosody in popular music,” in *submitted to Speech Prosody*, Poznań, Poland, 2018.

[16] D. Sacha, Y. Asano, C. Rohrdantz, F. Hamborg, D. Keim, B. Braun, and M. Butt, “Self Organizing Maps for the Visual Analysis of Pitch Contours,” in *Nordic Conference of Computational Linguistics*, 2015, pp. 181–189.

[17] Y. Asano, M. Gubian, and S. Dominik, “Cutting down on manual pitch contour annotation using data modelling,” in *Speech Prosody*, 2016, pp. 282–286.

[18] M. Gubian, F. Cangemi, and L. Boves, “Automatic and Data Driven Pitch Contour Manipulation with Functional Data Analysis,” in *Speech Prosody*, 2010, pp. 181–189.

[19] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, pp. 504–507, 2006.

[20] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, “Greedy layer-wise training of deep networks,” in *International Conference on Neural Information Processing Systems (NIPS)*, 2006, p. 153–160.

[21] D. Arpit, Y. Zhou, H. Ngo, and V. Govindaraj, “Why regularized auto-encoders learn sparse representation ?” in *International Conference on Machine Learning (ICML)*, Poznań, Poland, 2016.

[22] M. A. Ranzato, Y. l. Boureau, and Y. L. Cun, “Sparse feature learning for deep belief networks,” in *International Conference on Neural Information Processing Systems (NIPS)*, 2007, p. 1185– 1192.

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[24] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

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

[26] D. E. Rumelhart, G. E. Hinton, and v Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, p. 533–536, 1986.

[27] W. Fisher, “Tsylb Syllabification Package,” 1996. [Online]. Available: ftp://jaguar.ncsl.nist.gov/pub/tsylb2-1.1.tar.Z

[28] N. Obin, F. Lamare, and A. Roebel, “Syll-O-Matic: an Adaptive Time-Frequency Representation for the Automatic Segmentation of Speech into Syllables,” in *International Conference on Acoustics, Speech, and Signal Processing*, Vancouver, Canada, 2013.

[29] A. Camacho, “SWIPE: A Sawtooth Waveform Inspired Pitch Estimator for Speech and Music,” PhD. Thesis, University of Florida, 2007. [Online]. Available: http://www.cise.ufl.edu/~acamacho

[30] P. J.Rousseeuw, “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis,” *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

[31] P. D. Allison, *Multiple Regression: A Primer*. London: Sage Publications, 1998.