A Deep Paradigm for Articulatory Speech Representation Learning via Neural Convolutive Sparse Matrix Factorization

Anonymous ACL submission

Abstract

Most of the research on data-driven speech representation learning has focused on raw audios in an end-to-end manner, paying little attention to their internal phonological or gestural structure. This work, investigating the speech representations derived from articulatory kinematics signals, uses a neural implementation of convolutive sparse matrix factorization to decompose the articulatory data into interpretable gestures and gestural scores. By applying sparse constraints, the gestural scores leverage the discrete combinatorial properties of phonological gestures. Phoneme recognition experiments were additionally performed to show that gestural scores indeed code phonological information successfully. The proposed work thus makes a bridge between articulatory phonology and deep neural networks to leverage interpretable, intelligible, informative, and efficient speech representations.

1 Introduction

Research on speech representation learning has been dominated by deep learning in recent years (Latif et al., 2020). The goal of speech representation learning is to optimize both the performance of the model architectures and the interpretability of the learned representations. As there is growing demand of real-life applications of speech interfaces (Herff and Schultz, 2016), the performance is emphasized to a larger extent, enabling human-machine interactions highly accurate and robust. Consequently, in most of these works the interpretability of representations has not been explored to an equivalent extent, which is one of the most significant bottlenecks that keeps the speech research from going farther. In general, speech representations need to be better understood and developed.

People usually represent speech via audio because human perceive speech through hearing and audio is cheap to record, collect and process. However, speech processing is quite a lot different from audio processing. It might not need any evidence to indicate that any information that can be perceived via human can be perceived anywhere from source to destination. Perceiving the speech signal from the source and leveraging how it is produced are the most straightforward way to interpret it. The speech signal is the result of respiratory, phonatory and articulatory processes that generate the perceivable acoustic resonances to encode an intended linguistic message (MacNeilage, 2010). In that sense, perceiving the speech signal from articulatory data is a preferred way to derive interpretable, natural and robust speech representations.

The framework of articulatory phonology (Browman and Goldstein, 1992) has offered a lawful approach to modeling the relation between phonological representations as a set of discrete compositional units, or *gestures*, and the variability in time that derives from variation in the activation of the gestures in real-time: the magnitude of their activation, and the temporal intervals of activation as represented in *gestural scores*. However, the gestures and gestural scores of particular utterances have never been estimated in a completely data-driven manner. (Ramanarayanan et al., 2013) utilized the convolutive sparse non-negative matrix factorization (CSNMF) to decompose the non-negative articulatory data into the gestures and gestural scores, both of which are pretty much interpretable. The downsides of such method are that all the training utterances have to be concatenated into a large matrix, resulting in both memory and training efficiency issues. Additionally, such a model is not compatible with the modern deep learning based speech models so that it is challenging to perform end-to-end training on articulatory data.

To handle the aforementioned problem, (Smaragdis and Venkataramani, 2017) proposed an auto-encoder based model to replace non-negatve matrix factorization for speech separation task. Inspired by this work, we propose a convolutional

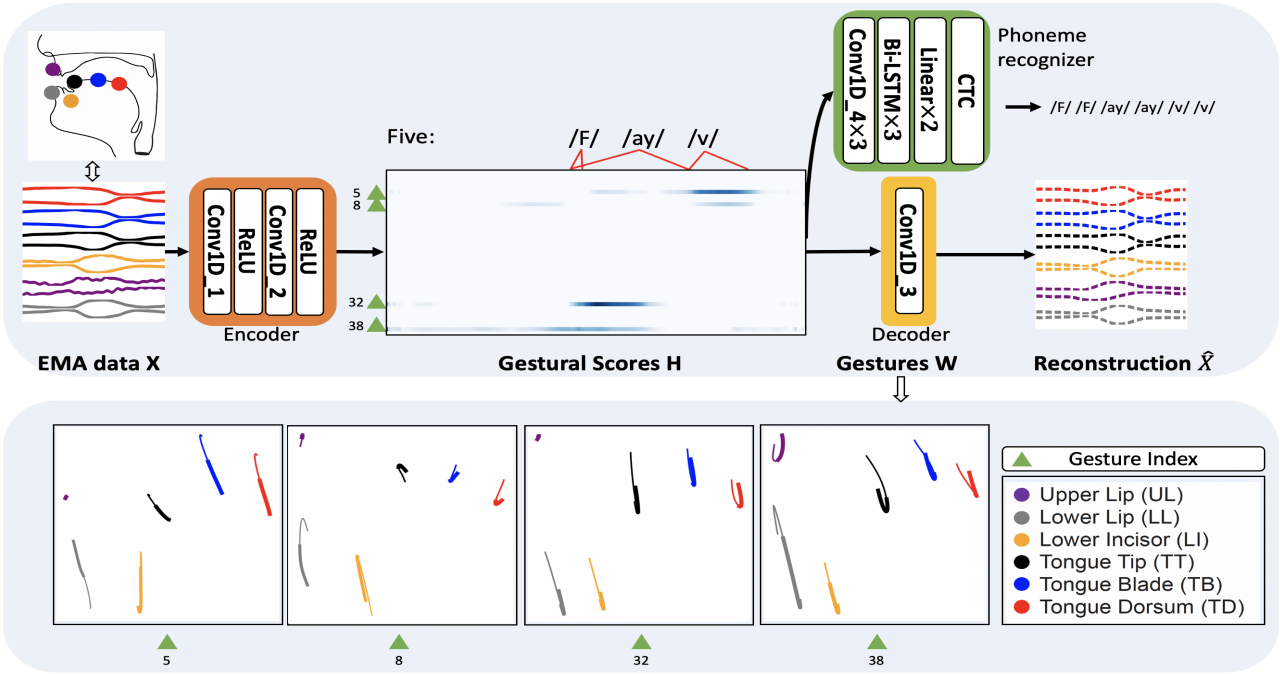

Figure 1: Neural Convolutive Matrix Factorization to Interpret Gestures and Gestural Scores.The panel in the bottom visualizes four activated gestures in the example utterance for *"Five"* . The gestures capture the moving patterns of articulators. Each moving pattern goes from thinner line to thicker line capturing about 200ms.

auto-encoder as the neural implementation of convolutive matrix factorization. Such auto-encoder based matrix factorization method is compatible with modern deep neural network and the batch-wise optimization improves the convergence rate to the huge extent. Under such framework, the articulatory signal is decomposed into *gestures* and *gestural scores* which are still interpretable. The gestural scores are the learned articulatory speech representations and are constrained to be sparse. In the last stage, the phoneme recognition experiments were performed to show that the learned gestural scores are also intelligible and consistent in time domain. All the experiments are performed using MNGU0 EMA (Electromagnetic midsagittal articulography) (Richmond et al., 2011) corpus. The intention is that the proposed work could bridge the gap between explainable articulatory phonology and modern deep neural networks to deliver interpretable, intelligible, informative, and efficient speech representations.

2 Proposed Methods

2.1 Neural Convolutive Sparse Matrix Factorization

Denote EMA data as $X \in \mathbb{R}^{C \times t}$, where (C,t) is (number of channels, segment length). By convolutive matrix factorization (Ramanarayanan et al., 2013):

$$X \approx \Sigma_{i=0}^{T-1} W(i) \cdot \overrightarrow{H}^{i} \quad (1)$$

$W \in \mathbb{R}^{T \times C \times D}$ is gestures and $H \in \mathbb{R}^{D \times t}$ is gestural scores, where D is number of gestures and T is the kernel size. $\overrightarrow{H}^{i}$ indicates that i columns of H are shifted to the right.

It is observable that Eq. 1 is actually the 1-d convolution with kernel W and input matrix H. By auto-encoder matrix factorization (Smaragdis and Venkataramani, 2017), H should be the hidden representation derived from the encoder which takes the pseudo-inverse of W as parameters. However, calculating the pseudo-inverse of high dimensional matrix is challenging. We experimentally justified that the encoder can be any types of neural networks with any number of layers. The proposed neural convolutive sparse matrix factorization is formularized as follows:

$$H = max(f(X), 0) \quad (2)$$

$$\hat{X} = W \odot H \quad (3)$$

where $f(.)$ denotes any type of neural network. In the original non-negative matrix factorization problem, all components (X, W, H) have to be non-negative. However, in such neural implementation, only H is required to be non-negative so that the gestures are always additive. There is no constraint for W and X.

2.2 Loss Objectives

There are a couple of items in the loss function. The first one is the reconstruction loss, which is L2 loss. The second one is sparseness. According to (Hoyer, 2004), the sparseness of a vector is defined as:

$$S(H_i) = \frac{\sqrt{n} - \frac{L_1(H_i)}{L_2(H_i)}}{\sqrt{n} - 1} \quad (4)$$

where H_i is the i-th row of H. L_1 and L_2 denote L_1 norm and L_2 norm respectively. n is the length of the vector. The sparseness of gestural score matrix is shown as below:

$$S(H) = \frac{1}{D}\Sigma_{i=1}^{D} S(H_i) \quad (5)$$

The third term is the entropy of the sparseness, denoted as:

$$E(H) = \frac{1}{D}\Sigma_{i=1}^{D}(-S(H_i)\log(S(H_i))) \quad (6)$$

It should be noticed that the sparseness cannot control the number of gestures that are activated. For instance, the H matrix with only one gesture activated for a long time interval might have the same sparsity with the matrix with multiple gestures activated for shorter time intervals. Typically we expect that a proper number of gestures should be activated. More intuition can be checked in Appendix A. We introduce two balanced factors λ_1 and λ_2 to limit both sparsity and entropy to a certain range. For EMA resynthesis task, the loss function is shown in Eq. 7, where $\mathbb{E}_X$ means the loss is computed by taking the average in the mini-batch.

$$L_{res} = \mathbb{E}_X[||X - \hat{X}||_2 - \lambda_1 S(H) + \lambda_2 E(H)] \quad (7)$$

For phoneme recognition experiments, CTC (Graves et al., 2006) loss L_{CTC} is used. For joint resynthesis-phoneme recognition task, the loss function is shown as in Eq. 8, where λ_3 is a balanced factor.

$$L_{joint} = L_{res} + \lambda_3 L_{CTC} \quad (8)$$

3 Experiments

3.1 Dataset

MNGU0 EMA (Electromagnetic midsagittal articulography) (Richmond et al., 2011) dataset is used in this work. There are in total 1263 utterances recorded from one single speaker. Details can be checked in Appendix B. The Mel-Spectrogram is used as acoustic feature with the framing configuration of 25ms/16ms and feature dimension of 80. The unaligned phonemes extracted from text transcriptions via the CMU pronouncing dictionary[1], are used as labels for phoneme recognition task. The train/test split is 8:2, which is the same for all experiments.

3.2 Tasks and Evaluation Methods

We perform two sets of experiments: (i) EMA Resynthesis. By resynthesizing the EMA data, we extract, visualize and interpret the gestures and gestural scores. The reconstruction loss (L2) averaged over all test samples is used to measure the *informativeness* of gestural scores (Saxe et al., 2019). The sparsity defined in Eq. 5 is used to measure the *efficiency* of gestural scores. The *interpretability* of gestures and gestural scores is evaluated by subjective analysis. (ii) Phoneme Recognition (PR). PER (Phoneme Error Rate) is used as metric for this task. PR on EMA is performed to measure the *intelligibility* of EMA data. PER on melspectrogram is performed to measure the *intelligibility gap* between articulartory and acoustics data. Lastly, the joint training of EMA resynthesis and phoneme recognition on gestural scores is performed to measure both the *intelligibility* (Lakhotia et al., 2021) and the *consistency* of learned sparse speech representations. Considering that EMA is not able to capture the difference between voiced and voiceless phones, we also relabel the phoneme sequence by assigning the same label to the phonemes with the same articulatory representation in EMA[2], and compute PER on new labels. We call the latter metric as PER-V, which is reported for all PR experiments.

3.3 Model Architectures

The overall model backbone is shown in Fig. 1. The encoder takes EMA data X in and outputs the gestural scores H. The decoder takes H in and resynthesizes EMA data $\hat{X}$. For independent phoneme recognition or joint resynthesis-CTC experiments, the phoneme recognizer takes EMA, melspectrogram or H in and predicts the alignment. Beamsearch algorithm is used for decoding with beam width of 50 in phoneme recognition task. Details can be checked in Appendix C.

[1] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[2] Specifically,these tuples are expected to have the same articulatory labels: (p,b,m), (t,d,n), (ch,jh), (f,v), (sh,zh), (k,g,ng), (s,z), (th,dh)

3.4 Implementation Details

For EMA resynthesis experiments, we randomly extract a segment with fixed length of 300 frames as the input of model for each iteration. For phoneme recognition experiments, the full utterance is taken as input. The training details can be checked in Appendix D. For the loss function in Eq. 7 and Eq. 8, we set $\lambda_1 = \lambda_2 = 10$ and $\lambda_3 = 1$. For resynthesis and resynthesis-CTC experiments, we explore different values of number of gestures: 20, 40, 60 and 80 as ablation studies. The results of EMA resynthesis, joint resynthesis-CTC and independent PR on EMA and melspectrogram are recorded in Table. 1 and Table. 2 respectively. To interpret the gestures and gestural scores, a random utterance is taken as input ("Five" in this example) to the encoder-decoder framework and we visualize the gestural scores as well as activated gestures, as shown in Fig. 1.

Table 1: Resynthesis and Resynthesis-CTC

#gestures	20	40	60	80
Resynthesis				
Rec Loss%	27.16	25.17	24.17	22.99
Sparsity(H)%	94.10	94.50	94.17	94.90
Resynthesis-CTC				
Rec Loss%	24.70	19.65	18.95	17.72
Sparsity(H)%	92.90	92.54	93.10	92.50
PER %	20.75	**14.10**	15.44	15.71
PER-V %	16.55	**11.02**	11.88	12.09

Table 2: PER on EMA and Melspec

Feature	EMA	Melspec
PER %	13.27	7.54
PER-V %	10.24	6.18

3.5 Discussion

We discuss the results in terms of four aspects of the learned gestural scores: (i) **Informativeness**. Lower reconstruction loss shows that the gestural scores are more informative. By making the comparison between the input EMA and synthesized EMA, we empirically observe that the reconstruction loss that is below 40% would not loss too much information. As shown in Table. 1, the larger the number of gestures, the more informative the gestures are. (ii) **Intelligibility and Consistency**. Based on Table. 2, EMA gives higher PER and PER-V than melspectrogram because it captures the information that is limited and discrete in space. PER-V of EMA is lower than PER, which is consistent to the fact that EMA is not able to differentiate voiced and voiceless phones. Based on Table. 1, when number of gestures is 40, both PER and PER-V are comparable to the results obtained from EMA, which shows that gestures scores are intelligible and consistent in time dimension. Note that when increasing the number of gestures, the PER is not always decreasing, indicating that the intelligibility is not always positive correlated to the informativeness. iii) **Efficiency**. Based on Table. 1, when number of gestures is 40, the sparsity of gestural scores is 0.9254, showing 90% of the space is saved without a heavy degradation of PER. (iv) **Interpretability**. Subjective evaluation was performed. As shown in Fig. 1, when "Five" is taken as input, four gestures(5,8,32,38) are activated with different activation intervals. /F/ is expected to be produced by a raising of the Lower Lip which is here accomplished by gesture 5. The same gesture also lowers the the tongue and jaw, which is expected for the beginning of the diphthong /AY/. The fact that this pattern is contributing to both the consonant and the vowel is sensible, as word-initial consonants and the following vowels are known to be initiated at roughly the same time (Goldstein et al., 2006). Gesture 32 is also strongly activated during the time of the beginning of the diphthong and it lowers all of the markers on the lower surface, again as expected for the beginning part of the diphthong /AY/. The second part of the diphthong involves raising of the jaw and tongue tip, and this is accomplished here by gesture 8 that is active near the end of the word. Gesture 5 is also engaged at the end of the word that raises the lower lip for the /V/.

4 Conclusion

This work proposes a neural convolutive sparse matrix algorithm which decomposes the EMA data into gestures and gestural scores. The learned representations a.k.a gestural scores are informative, intelligent, consistent, efficient and interpretable. This method bridges the gap between articulatory phonology and deep learning techniques. Hopefully the proposed work could become a paradigm that benefits both the downstream explorations that are helpful for patients with vocal cord disorders and the explorations in industrial applications towards roboust, controllable and generalizable speech synthesis.

References

Catherine P Browman and Louis Goldstein. 1992. Articulatory phonology: An overview. *Phonetica*, 49(3-4):155–180.

Louis Goldstein, Dani Byrd, and Elliot Saltzman. 2006. The role of vocal tract gestural action units in understanding the evolution of phonology. *Action to language via the mirror neuron system*, pages 215–249.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Christian Herff and Tanja Schultz. 2016. Automatic speech recognition from neural signals: a focused review. *Frontiers in neuroscience*, 10:429.

Patrik O Hoyer. 2004. Non-negative matrix factorization with sparseness constraints. *Journal of machine learning research*, 5(9).

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kushal Lakhotia, Evgeny Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Adelrahman Mohamed, et al. 2021. Generative spoken language modeling from raw audio. *arXiv preprint arXiv:2102.01192*.

Siddique Latif, Rajib Rana, Sara Khalifa, Raja Jurdak, Junaid Qadir, and Björn W Schuller. 2020. Deep representation learning in speech processing: Challenges, recent advances, and future trends. *arXiv preprint arXiv:2001.00378*.

Peter F MacNeilage. 2010. *The origin of speech*. 10. Oxford University Press.

Vikram Ramanarayanan, Louis Goldstein, and Shrikanth S Narayanan. 2013. Spatio-temporal articulatory movement primitives during speech production: Extraction, interpretation, and validation. *The Journal of the Acoustical Society of America*, 134(2):1378–1394.

Korin Richmond, Phil Hoole, and Simon King. 2011. Announcing the electromagnetic articulography (day 1) subset of the mngu0 articulatory corpus. In *Twelfth Annual Conference of the International Speech Communication Association*.

Andrew M Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox. 2019. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020.

Paris Smaragdis and Shrikant Venkataramani. 2017. A neural network alternative to non-negative audio models. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 86–90. IEEE.

Appendices

A Sparse Entropy Loss

Fig. 2 gives an intuition of entropy loss. All three H matrices have the same sparsity. If the entropy is pretty low, only one gesture is activated, which leaves many other gestures unused. If the entropy is pretty high, some of activated gestures are redundant, which makes the gestural score less explainable.

(a) Low Entro (b) Medium Entro (c) High Entro

Figure 2: Three types of gestural score matrices with the same sparsity but different entropy values.

B Dataset

MNGU0 EMA (Electromagnetic midsagittal articulography) (Richmond et al., 2011) dataset is used in this work. There are in total 1263 utterances recorded from one single speaker. During the recording, six transducer coils were placed in the midsagittal plane at the upper lip, lower lip, lower incisors, tongue tip, tongue blade and tongue dorsum to record the coordinates (x and y) of their positions, and thus each EMA data frame takes 12 coordinates, as shown in Fig. 1. The sampling rate of EMA is 200 Hz.

C Model Details

Table 3: Model Configurations

Module Name	Block name	Configurations*
Encoder	Conv1d_1	(15,64)×1
	Conv1d_2	(5,D)×1
Decoder	Conv1d_3	(41,C)×1
Phoneme Recognizer	Conv1d_4	(5,64)×3
	Bi-LSTM	(256)×3
	Linear	(128)×2

* For Conv1d Block, it is (kernel size, output channels)×# of layers. For Bi-LSTM and Linear layers, it is (output dimension)×# of layers.

The configurations of each module in Fig. 1 are shown in Table. 3, where the number of gestures D is a hyperparameter and C is 12. For all convolutional layers, the stride is 1 and paddings are made so that the output length keeps the same across layers. Batchnorm1D (Ioffe and Szegedy, 2015) is applied after each convolutional layer.

D Training Configuration

All experiments were trained on Nvidia Tesla V100 GPU. It takes one GPU hour to run a single EMA resynthesis experiment and 5GPU hours to run phoneme recognition as well as resynthesis-CTC experiments. Optimizer is Adam (Kingma and Ba, 2014) with the initial learning rate of 1e-3, which is decayed every 5 epoches with a factor of 5. Weight decay is 1e-4. Batchsize is 8. The weights of decoder (gestures) are initialized by the centers from a kmeans algorithm: Slide the window of size 41 with a stride of 1 on EMA kinematics data, concatenate all 41 vectors into a supervector and perform kmeans on all supervectors.