What does the Canary Say? Low-Dimensional GAN Applied to Birdsong

Silvia Pagliarini, Nathan Trouvain, Arthur Leblois, Xavier Hinaut

What does the Canary Say? Low-Dimensional GAN Applied to Birdsong

Silvia Pagliarini[1,2,3], Nathan Trouvain[1,2,3], Arthur Leblois[3*], and Xavier Hinaut[1,2,3*].
1. INRIA Bordeaux Sud-Ouest, Bordeaux, France. 2. LaBRI, Université de Bordeaux, Institut Polytechnique de Bordeaux, Centre National de la Recherche Scientifique, UMR 5800, Talence, France. 3. Institut des Maladies Neurodégénérative, Université de Bordeaux, France, Centre National de la Recherche Scientifique, UMR 5293, Bordeaux, France.

***Abstract*—The generation of speech, and more generally complex animal vocalizations, by artificial systems is a difficult problem which has recently been addressed using various techniques in artificial intelligence. Generative Adversarial Networks (GANs) have shown very good abilities for generating images, and more recently sounds. The usability of a GAN generating a vocal repertoire relies in part on our understanding of the representations of the various sounds in the GAN latent space. Here, we aim to test the ability of WaveGAN to produce a set of canary syllables and constrain the latent space to a small dimension. We trained WaveGANs with varying latent space dimensions (from 1 to 6) on a large dataset of canary syllables (16000 renditions of 16 different syllable types). The sounds produced by the generators are identified and evaluated by a RNN-based classifier trained on the same dataset. This quantitative evaluation is paired with a qualitative evaluation of the GAN output spectrograms across GAN training epochs and latent dimensions, comparing multiple instances of the training for each condition. Altogether, our results show that a latent space of dimension 3 is enough to produce a varied repertoire of sounds of quality often indistinguishable from real canary ones, spanning all the types of syllables of the dataset. Importantly, we show that the 3-dimensional GAN generalizes by interpolating between the various syllable types. We rely on UMAP representations to qualitatively show the similarities between the training data and the generated data, and between the generated syllables and the interpolations produced. Exploring the latent representations of syllable types, we show that they form well identifiable subspaces of the latent space. This study provides tools to train simple sensorimotor models, as inverse models, from perceived sounds to motor representations of the same sounds. Both the RNN-based classifier and the small dimensional GAN provide a way to learn the mappings of perceived and produced sounds.**

***Index Terms*—Generative adversarial networks, latent space, sound generation, birdsongs.**

I. Introduction

The acoustic variety of speech sounds across speakers, rates and contexts makes speech composition difficult to apprehend. Indeed, when different speakers produce the same phonem, the acoustics (i.e. formant frequencies) vary [1], making speech highly variable. Faster speech acoustics differ from slower speech acoustics, and contexts variations can determine a change in the acoustic features [2]. Behavioral and the neuroanatomical similarities between songbirds and humans [3][4] make songbirds a particularly tractable model for a *reduced* version of the vocal learning process in humans, enabling to test basic mechanistic hypotheses [5]. Among the many songbird species, canaries have a large highly variant repertoire and their songs are characterized by a complex syntax with long-time dependencies [6]. These properties make canary songs a reasonable middle ground between human speech and simpler birdsong.

The development of vocal learning models (for speech or birdsong) require the implementation of a realistic vocal production apparatus. It remains unclear whether known learning algorithms can be used to learn high-dimensional and continuous tasks, such as producing a given sound [7]–[9].

Latent variable models (or *latent space models*) are a class of models that allows to embed high dimensional data into a low dimensional representation (called *latent space*) where nearby points have similar properties [10]. Despite the low-dimension of their latent representation, the generated data can realistically reproduce the real data with a diversity of outputs, including elements not present in the training dataset [10]. By interpolating between two points in the latent space associated with 2 different outputs, a continuous set of intermediate realistic outputs can be generated [11]. These properties (low dimension, diverse output, continuous representation) highlight the usefulness of latent space models to obtain a tractable representation of a given dataset.

Generative Adversarial Networks (GANs) are an example of generative models which enable to represent high-dimensional distributions in a latent space. The main components of a GAN are a generator, that tries to reproduce a target distribution (e.g. images) given random inputs, and a discriminator that estimates the probability that the data generated by generatoris coming from the data distribution or not [12]. The "latent space" of a GAN is the input space of the generator after training. GANs have been used to produce complex datasets such as images, sounds, music, speech and, to a lesser extent, birdsongs [13]–[15]. Donahue et al. [15] trained their model, called WaveGAN, on a speech, drums and piano, and on wild recordings from several bird species. The promising but noisy results on a wide and highly variable dataset of birdsongs determined our choice to study the performance of the WaveGAN generator.

In this paper, we trained a Wesserstein GAN with Gradient Penalty (WGAN-GP) [16] using the WaveGAN [15] setup on an adult canary dataset [17]. The main objective of this study

* Corresponding authors who co-supervised the study.
Manuscript draft on May 28th, 2021.

is to test the ability of this GAN to produce realistic canary syllables (a vocal repertoire akin to the vocalizations of a single adult canary) for different conditions, varying the latent space dimension and the size of the training dataset. On one hand, we are interested in finding the minimal latent space dimension that allows to reproduce real songs fluctuations. Having a low dimensional latent space does not reduce the computational costs of GAN training (or marginally if it does). Thus, in general, there is no need for such a low-dimensional latent space in deep learning studies. At the same time, a low-dimensional latent space is easier to represent, explore and analyze. In addition, a low-dimensional space would require less weights parameters to be learned by a neural network, and thus less prone to overfitting. Although a greater number of motor dimensions would not be a problem to motor babbling while learning [18], for robot control, the learning of the inverse kinematics is more difficult when the degree of freedom in the motor space is much higher than the number of controlled dimensions (i.e., the number of possible sensory outputs) [19]. Thus, a low motor dimension should facilitate the learning when used in sensorimotor models [9], where an imitative agent wants to learn to reproduce the sounds it perceives: it has to learn the inverse model from perceived sounds to the corresponding motor commands.

On the other hand, we are interested in exploring the capability of the generator to reproduce a good output when the network has been trained with datasets of limited size. Indeed, having a lower amount of training data would speed up the computational time to train the generator. We also explore the latent space obtained in order to measure (quantitatively and qualitatively) the accuracy and diversity of the produced syllables, the continuity of the latent space (i.e. the ability of the generator to generalize), and the structure of the latent representations.

Section II contains an introduction to Generative Adversarial Networks (GANs), a brief mention to the models that are relevant for this study and the description of the architecture of the WaveGAN used here. Section III-A describes the procedure applied to sort previously recorded canary vocalizations and build the training dataset. In section III-B, we present the training dataset and detail the different training conditions we used to train WaveGAN. Section III-C introduces the metrics used to evaluate the performance of the generator. The tools used to evaluate the model are in Sections III-D and III-E. Section IV contains the results obtained in this study. The capacity of the generator to reproduce realistic samples is confirmed by quantitative and qualitative analysis. Section IV-A shows how such analysis works on the training dataset. Section IV-B shows the analysis we carried out to evaluate the ability of the generator to reproduce the whole repertoire. Section IV-C shows the organization the latent space. Section IV-D and Section IV-E show the quantitative analysis presented in Section IV-B applied to compare the generator performances when using different conditions for the latent space dimension, or for the size of the training dataset. Section V summarizes the results we obtained and explains the advantages and the limitations of the network. Moreover, we discuss how a generator model such the one investigated by this study could be used in future work within the framework of vocal learning models.

II. GAN BACKGROUND

A. Generalities about GANs

Recently, several generative models have been proposed, such as Wavenet [20], Variational autoencoders (VAEs) [21] and Generative Adversarial Networks (GANs) [22]. In particular, GANs have been introduced for the first time by Goodfellow et al. [22] as a novel class of machine learning frameworks. Two models, the generator model and the discriminator model, compete to become better at their objective. The generator model aims to be able to produce samples that come from the training data distribution, without having access to any information regarding them. On the contrary, the discriminator has access to the distribution of the training dataset, and should be able to discriminate between a sample which belongs to the training dataset and a sample generated by the generator model [23].

In the original formulation, the Jensen-Shannon divergence is used as a loss function [22]. Several authors proposed new GANs varying the model architecture or the loss function. For example, a more stable training has been obtained using deep convolutional neural networks for both the generator and the discriminator: this is the case for Deep Convolutional GAN (DCGAN) proposed by Radford et al. (DCGAN) [11]. Alternatively, Berthelot et al. [24] proposed Boundary Equilibrium GAN (BEGAN) where the discriminator is an autoencoder [25]. Arjovsky et al. used a DCGAN architecture in Wasserstein GAN (WGAN) [16] and improved it by modifying the loss function definition, such that it is robust to changes in the network architecture.

Firstly, the role of the discriminator has changed: it is not making anymore a direct choice to assess whether a sample is real or fake. Instead, the discriminator acts as a critic and provides the generator a loss allowing it to be trained until it reaches an optimum. An advantage of the critic is that it can't saturate and converges to a linear function, whereas the classic discriminator could learn too quickly, becoming not reliable [16]. Secondly, the loss is based on the computation of Wasserstein distance, which gives stability to the GAN and avoids mode collapsing [14]. Moreover, weight clipping is used to enforce the continuity of the loss function. Finally, WGAN has been improved by the introduction of a regularization parameter, usually called λ.

Gulrajani et al. [26] proposed the addition of a Gradient Penalty (GP) term in the loss function, driven by λ. GP penalizes any deviation of the gradient norm of the critic. This enables a faster training and less parameter tuning. This model is called WGAN with Gradient Penalty (WGAN-GP). Alternatively, Petzka et al. [27] proposed the addition of a Lipshitz Penalty (LP) term in the loss function, driven by λ. LP penalized only larger (> 1) deviations of the gradient norm (and not all of them, as GP does). For small values of λ WGAN-GP and WGAN-LP perform similarly, whereas for larger values of λ the performance of WGAN-GP are more λ-dependent [27].

MuseGAN [14] and WaveGAN [15] are two examples of application of WGAN-GP to achieve sound generation. Dong et al. [14] used a dataset of multi-track piano-rolls derived from the Lakh Midi Dataset (LMD) [28] and train WGAN-GP to generate multi-track sequences. Donahue et al. [15] trained WGAN-GP to generate music (piano, drums), speech (i.e. the Speech Commands Zero through Nine-SC09 dataset) and, to a lesser extent, birdsong (using a training dataset composed by wild recordings of different species).

B. WaveGAN

Dealing with a GAN generating sounds is different from generating images: Donahue et al. [15] highlighted this using Principal Component Analysis (PCA). Principal components capture gradient, intensity and characteristics of the edges for images, whereas principal components form a periodic basis that decompose the audio signal for waveforms [15]. Indeed, audio signals show more periodicity than images. Thus, a larger receptive field to process audio signals is introduced in WaveGAN. This is similar to what Van den Oord et al. [20] did in WaveNet, where dilated convolutions has been used to increase the effective receptive field of the model.

WaveGAN architecture is based on the architecture of DCGAN [11] which uses GANs for image synthesis. DCGAN [11] generator is a CNN where transposed convolution is used to upsample low-resolution feature maps into a high-resolution image. As for DCGAN, the generator and the discriminator of WaveGAN are CNNs.

Donahue et al. [15] used WGAN-GP[26] strategy during training. This strategy consists in the introduction of a gradient penalty term in the loss function, driven by the regularization hyperparameter λ. The advantages of using GP are a faster training and the avoidance of problems coming from weight clipping (used to force the gradient to stay below a certain threshold), which were present in the original formulation [26]. Please refer to the supplementary material (see Figure 18) for further details about the architecture of WaveGAN.

III. Methods

A. Data pre-processing

Canaries sing sequences of syllables organized in phrases: a phrase is a short segment of the song where the same syllable is repeated many times [6]. The initial dataset was composed of a repertoire of 27 classes of syllables organized in labeled phrases (i.e. each phrase was already assigned to a specific class), manually sorted from a set of recordings at $44100Hz$ from an adult canary [17]. Among the 27 classes, we focused on 16 classes in order to have only classes with enough samples: $A, B1, B2, C, D, E, H, J1, J2, L, M, N, O, Q, R, V$. We performed the following steps on each phrase:

1) We downsampled each phrase to a sampling rate of $16000Hz$.
2) From each phrase we made a first syllable selection tuning three parameters: the amplitude threshold, the minimal duration of the syllable and the duration of the gap (i.e., the silence between two consecutive syllables).
3) We applied to all the selected syllables a high-pass filter of order 5 to remove all frequencies below $700Hz$.

Figure 15 in the supplementary material shows a summary of the selection procedure for syllable $J2$. The automatic selection based on an amplitude threshold, the duration of the syllable and the duration of the gap, could fail to select correctly the syllable. We performed a visual inspection after selection on duration to filter misclassified and miscut elements. The resulting dataset contains 72155 syllables from the 16 classes. In supplementary material, Figures 16 and 17 show 100 examples of samples that have been selected from that class after the pre-processing.

B. Experimental setup

We selected a balanced training dataset in order to consider almost the same number of samples per syllable type. Among all the selected syllables, we used a subset of $16k$ syllables: $1k$ syllables per class. From now on, we will refer to this balanced dataset as the training dataset. Each syllable has been padded with silence to obtain recordings having a length of exactly 1 s. This step has been done to create a dataset resembling the speech dataset that was originally used to train WaveGAN [15]. We used the balanced training dataset to train both the classifier (described in Section III-D) and the network (described in Section II).

We used the original WaveGAN [15] setup to train the network. We used the original network architecture with gradient penalty option, with $\lambda = 10$ and Adam optimizer in the training phase. We used a batch size of 64 samples and we trained the discriminator 5 times more than the generator.

We tested different conditions for the latent space dimension (indicated as *ld*), varying it from $ld = 6$ to $ld = 1$. In the following, we will refer to one of these particular conditions using, for instance, 6-dimensional WaveGAN to refer to a GAN trained with a 6-dimensional latent space, or 5-dimensional WaveGAN to refer to a GAN trained with a 5-dimensional latent space. We tested different sizes for the training dataset. First, we trained WaveGAN using the complete dataset, then we reduced it by a factor of ~ 2 (i.e., $8k$ syllables) and finally by a factor of ~ 4 (i.e., $4k$ syllables).

We first trained 3 instances per latent space condition (keeping the dataset size fixed) and dataset size condition (keeping the latent space dimension fixed), to observe the performance of the generator across time. We selected from the availabe instances the best one: either overtraining is not present, or it does not happen too soon. We compared the results to choose the optimal combination of latent space dimension and dataset size. We then trained 10 instances of WaveGAN with the training dataset introduced at the beginning of this Section. For all the instances, we trained the network until epoch ~ 1000, saving the model every ~ 15 epochs. After training, we used the generator to produce new syllables at every saved epoch. We evaluated the generated data using both a quantitative and a qualitative measure, as described in Sections III-C1 and III-C2.

C. Evaluation

We evaluated the performance of the generator across epochs and across training conditions, in order to see the

optimal choice of parameters. As explained in Section III-B, after training, we used the generator to produce new syllables every 15 epochs and we used the classifier described in Section III-D to identify the generated syllables as elements of one class of the vocabulary. The vocabulary is the set of classes that the classifier knows, and in which it can classify each syllable. Accordingly with the type of evaluation we want to perform, we defined two vocabularies and trained two classifiers.

First, we used the balanced training dataset to train a classifier able to provide the probability of each sample belonging to each class of a vocabulary composed by the 16 classes of the repertoire. We refer to this model as *classifier-REAL*.

Then, we trained a classifier able to provide the probability of each syllable belonging to 21 classes: the 16 classes of the repertoire, a white noise (*WN*) class, an overtraining (*OT*) class, and three *EARLY* classes, respectively obtained from epochs 15, 30 and 45. To train the classifier to recognize the alternative unknown classes, in addition to the usual training dataset, we used three additional sets of generated samples. In summary, to train the classifier we used:

- *EARLY 15*: $1k$ samples of early generations, obtained after $\sim$ 15 epochs using two different instances of a 3-dimensional WaveGAN (respectively, 500 samples per instance);
- *EARLY 30*: $1k$ samples of early generations, obtained after $\sim$ 30 epochs using two different instances of a 3-dimensional WaveGAN (respectively, 500 samples per instance);
- *EARLY 45*: samples of early generations, obtained after $\sim$ 45 epochs using two different instances of a 3-dimensional WaveGAN (respectively, 500 samples per instance);
- *OT*: $1k$ samples obtained when two instances of a 3-dimensional WaveGAN reach overtraining (respectively, 500 samples per instance);
- *WN*: $1k$ samples of artificial white noise.

The two different instances used to define the classes above are instances *Ex 0* and *Ex 1* (where *Ex* stands for *example*) in Figure 34. We will refer to this classifier as *classifier-EXT* (where *EXT* stands for extended).

For simplicity of notation, we will call the set of unknown classes *X*. We define a class $x \in X$ as a class representing either white noise, or samples containing a lot of noise and, in general, resembling early productions of the generator (i.e., belonging to *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*).

1) ***Quantitative evaluation****:* For GANs, a quantitative measure is a measure which allows to understand if the model is able to reproduce a wide enough variety of samples [29], and provides a preliminary measure of whether or not the generated data resembles the real syllables. To observe the generator's performance across time we will describe how many classes it is able to produce and how many syllables per class it is able to produce in average. In this way, we studied the stability of WaveGAN across different instances of training. Moreover, we computed the Inception Score (IS) at several epochs of the training, we observed its evolution and we compared it with the IS obtained from the training dataset. IS has been proposed by Salimans et al. [13] as a quantitative measure to evaluate GANs: a pre-trained deep learning neural network model for image classification provides the probability of each image belonging to each class. This information is then summarized in the IS, which is defined as follows:

$$IS = exp(\mathbb{E}(KL(p(y|x)||p(y)))), \tag{1}$$

where KL stands for Kullback-Leibler divergence. Given a problem with N classes, $IS \in [1, N]$. IS provides both a measure of the quality and of the entropy of the generations, giving an idea of the generator ability to produce a wide set of new data [13]. The IS has been used by several authors [15][26][30] as a method of objective evaluation for GANs performance.

2) ***Qualitative evaluation****:* First, we based our qualitative analysis on spectrogram analysis. Then, we explored the latent space to study its structure and the continuity of the generations.

On the one hand, we computed the mean spectrogram of the generated data, and we compared it with the mean spectrogram of the dataset and with the repertoire. To obtain the mean spectrogram, for each class, we first aligned the syllables' envelope, then we computed the mean of the spectrograms of all the syllables belonging to that class. We observed the spatial organization of the data using Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) [31]. We applied UMAP to the spectrograms of the samples. Further details about the algorithm can be found in Section III-E.

On the other hand, we compared the spectrogram space with the latent space to observe the organization of the vectors with respect to the syllables they produce. We generated syllables for each small variation of the latent vector.

- **One component variation.** We selected a random latent vector $z \sim R^3([-1, 1])$ to generate a baseline syllable using a 3-dimensional WaveGAN after training. Then, we moved one by one the components of the vector by a variation step equal to $v_{step} = 0.05$. To explore critical points (i.e., where a bigger variation arises a sudden non-smooth change between two syllables) we moved one by one the components of the vector by a variation step equal to $v_{step} = 0.01$ and $v_{step} = 0.001$. Moreover, we used a step $v_{step} = 0.001$ to vary one by one each component of z between $[-1, 1]$ and we compared the generations obtained by each variation (one per component of the latent vector) with a set of $16k$ generated data from the same epoch.
- **Interpolation between two syllables.** We first selected 2 syllables s_1 and s_2 and their correspondent latent vectors z_1 and z_2, where $z_1, z_2 \in R^3([-1, 1])$. Then, we moved in the latent space from z_1 to z_2 using a variable step depending on the distance between the components of z_1 and z_2. That is, the step applied to each component i is

$$step[i] = \begin{cases} \frac{z_1[i]-z_2[i]}{N_{step}} & z_1[i] > z_2[i], \\ -\frac{|z_1[i]-z_2[i]|}{N_{step}} & otherwise, \end{cases} \tag{2}$$

where N_{steps} is the number of steps. We used $N_{steps} = 1000$. We used *classifier-EXT* to identify the syllables and see which class is assigned to the transition between s_1 and s_2. We used the UMAP representation to compare the variations with a bigger set of generated data.

Finally, we used human judgment to provide an additional qualitative evaluation. We asked two humans to participate in a syllable recognition test organized as described below.

- **Human training phase.** Recognition of a sample of 100 syllables from the training dataset: for the first 50 the person is authorized to look at the repertoire to classify the syllables. After each guess, the person can also look at the correct answer to continue learning to classify syllables. Then, the last 50 syllables have to be recognized. No help is allowed here, and no correct answer can be seen. In the training phase, the available classes are the 16 classes of the repertoire.
- **Human testing phase.** Recognition of a sample of 200 syllables generated after training, without the possibility to look at the repertoire. In the testing phase, the available classes are the 16 classes of the repertoire plus a general *X* class which, ideally, represents the alternative unknown classes (three *EARLY* classes, *OT* class and *WN* class) recognized by the classifier.

Then, we computed the proportion of agreement without considering the chance agreement (i.e. percentage of agreement that would have occurred anyway) using Cohen's kappa coefficient [32], [33]. If $\kappa_{Cohen} = 0$, then the agreement is equal to chance agreement. Alternatively, $\kappa_{Cohen} \in (0, 1]$ represents a positive agreement and $\kappa_{Cohen} = 1$ represents a perfect agreement between two judges [32]. We computed κ_{Cohen} for each of the participant with respect to *classifier-EXT* and for each couple of participants.

D. Classifier

The two classifiers (*classifier-REAL* and *classifier-EXT*) used during the evaluation phase (described in Section III-C) are *Echo State Networks* (ESNs) [34], a type of artificial recurrent neural network (RNN). ESNs are part of the Reservoir Computing paradigm) [35] that use random RNNs and train only the output layer. ESNs are often considered as temporal Support Vector Machines (SVMs): they work in a similar way by embedding input data into a high-dimensional space using non-linear transformations. However, unlike SVMs, ESNs are designed to manipulate sequential data and are relevant candidates for sound classification [36].

The classifiers were fed with Mel-Frequency Cepstral Coefficients (MFCCs) representations of the syllables, a low dimensional spectral representation of sound. We extracted 20 MFCCs features per time step, one time step being defined as the result of a spectral analysis window of $64ms$, using a Hanning window (often called *window width*; param. *win_length* in *librosa* library) applied to the $16kHz$ audio signal, with a $32ms$ jump between each frame (often called *frame stride*; param. *hop_length*) Because the GAN generated syllables tend to have higher amplitude than the real ones, we used only the first and the second derivative of the extracted MFCCs to remove any influence of the signal amplitude in the representations. Otherwise, the amplitude difference would bias the classifiers decisions, as it would be artificially easier to separate real samples from generated samples only by comparing the average power of the signals. First and second derivatives of the MFCC signal are also known to be good representations of vocal signals, as they give relevant clues on the temporal dynamics of the vocalizations.

The ESNs trained on the classification tasks [36] are built using *RerservoirPy* Python library [37][1] and are described by the following equations:

$$\begin{cases} x(n) = (1-\alpha)x(n-1) \\ \qquad\quad +\alpha \tanh\left(W_{in}u(n) + Wx(n-1)\right) \\ y(n) = W_{out}x(n) \end{cases} \tag{3}$$

where $u(n)$, $x(n)$ and $y(n)$ are respectively the input features, the internal state of the network and the output vector at time step n. W_{in} stores the connection weights between the inputs and the units of the network. These weights are sampled from a discrete bi-modal distribution, i.e. are randomly chosen from the set $\{1, -1\}$. The proportion of non-zero connection weights is fixed to 10%. W_{in} is defined in $\mathrm{R}^{N\times I}$, where N is the number of recurrent units and I is the dimension of the input. In our case $I = 41$, with input features being 20 derivatives of MFCCs, 20 second derivatives of MFCCs, and a constant bias equal to 1. Each set of features is scaled by multiplying the corresponding connection weights in W_{in} by a constant. The first derivatives and the bias are scaled by a factor 1, and the second derivatives are scaled by a factor 0.7. W stores the connection weights of the recurrent units. These weights are sampled from a standard normal distribution, with a proportion of non-zero connections between neuronal units fixed to 10%. W is defined in $\mathrm{R}^{N\times N}$, with N equal to 1000. W is scaled using a factor equal to a fixed *spectral radius* of 0.5 divided by the largest absolute eigenvalue of W. The α parameter, called *leaking rate*, is set to 0.05. It controls the time constant of the ESNs and allows information from past internal states to be fed to future internal states. All connection weights defined in W_{in} and W are fixed during the training phase of the ESNs, as opposed to machine learning algorithms using gradient descent. Only the $W_{out} \in \mathrm{R}^{N\times V}$ readout matrix is learned during training, where V is the output dimension corresponding to the vocabulary size used to classify the input features, i.e. $V = 16$ for *classifier-REAL* and $V = 21$ for *classifier-EXT*. The readout weights are learned using a linear regression between all the internal states x generated from the inputs and all the expected values of the output y. A *L2* regularization coefficient of value 10^{-8} is applied during the linear regression: thus this corresponds to a ridge regression.

The classifiers then output a vector $\hat{y}(n)$ of dimension V representing its activation for each time step n and for each category of syllable in the vocabulary. Then, these output activities are summed up over the whole sequence of MFCC frames. A softmax operation finally provides a probability

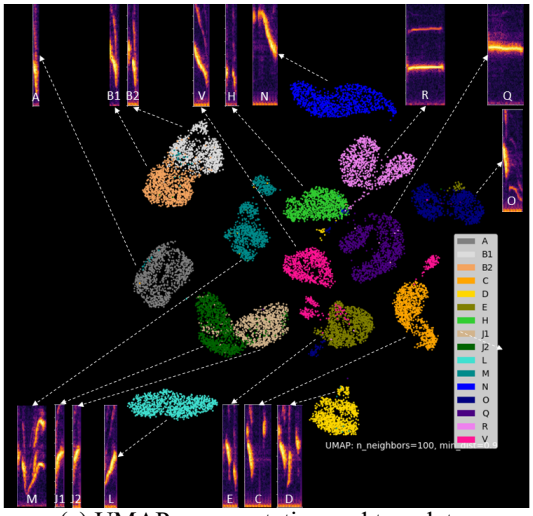
[1] https://github.com/reservoirpy/reservoirpy

distribution representing the chance for the syllable to belong to one of the classes of the vocabulary.

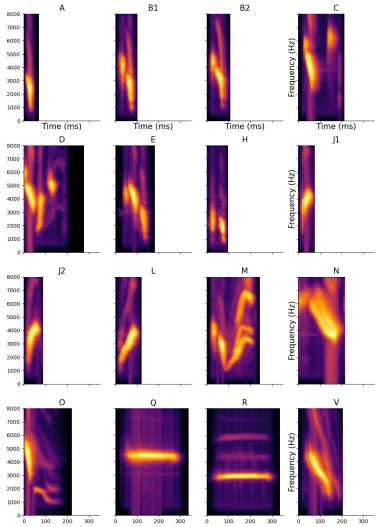
A further analysis of the robustness of the classifiers can be found in Appendix III-B.

E. Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP)

Similarly to t-SNE, Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) [31] is a dimension reduction technique. It can be used to perform non-linear dimension reduction. With respect to t-SNE, it has a higher computational power (i.e., it is faster). Moreover, it has the advantage of not being local (i.e. it takes into account the distance between points that are far in the space). This enables a synthetic, clear and simple representation of the original manifold. The power of using UMAP representation to reduce the complexity of songbirds spectrograms has first been shown by Sainburg et al. [38]: an exhaustive set of representations show how UMAP helps in representing not only the repertoire of a single species but also different species at the same time. The result is a two-dimensional representation. The axis are not meaningful to identify a discriminant feature (i.e., they do not represent, for example, the pitch of the syllable, or another syllable-related feature). Insteat, they are hyperparameters that well represent the given dataset: for instance, the size of the neighborhood used to estimate the manifold structure of the data and the minimum distance apart that points are allowed to be. The tuning of these hyperparameters allows to obtain a more or less local representation of the data (where a higher size of the neighborhood translates in a more local representation) and to pack or not pack together the points in the clusters (where a higher distance allows a more sparse representation).



(a) UMAP representation and template.



(b) Mean spectrogram.

Fig. 1. **Repertoire.** (a) UMAP [31] representation of the training dataset. Each cluster represents a class of the repertoire and a template syllable of each class is highlighted with the corresponding spectrogram (an arrow connects each cluster to the corresponding template). Syllables *B1* (cream cluster) and *B2* (light orange cluster) and syllables *J1* (dark green cluster) and *J2* (light brown cluster) lie very close, indeed the syllables are very similar and correspond to different speed of repetitions when embedded within a canary phrase. They will be merge in the following experiments. (b) Mean spectrogram computed after envelope alignment of the waveforms. To obtain the UMAP representation and the mean spectrogram we used $1k$ syllables per class (i.e., the training dataset) and their real labels. No classifier has been applied to assign each syllable to the correct class.

IV. Results

A. Analysis of the training dataset

This section describes the training dataset and highlights the performance of our evaluation metrics on it. The majority of the samples belong to 16 independent clusters in the Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) [31] representation (Figure 1a). Besides, the representation shows similarities between syllables *B1* and *B2* and syllables *J1* ad *J2*: the clusters of syllables *B1* (cream cluster) and *B2* (light orange cluster) and the clusters of syllables *J1* (dark green cluster) ad *J2* (light brown cluster) lie very close. These similarities can be noticed also in the spectrogram representation (Figure 1b). For this reason, in further representations using UMAP syllables *B1* and *B2*, *J1* ad *J2* have been grouped, respectively, into syllable *B* (keeping the light orange color to represent the cluster) and *J* (keeping the dark green color to represent the cluster) because of their high similarity. Each template shown in Figure 1a can be compared with the correspondent mean spectrogram (Figure 1b).

The level of confidence of the classifier in making the correct assignment can be represented using the confusion matrix relative to the predictions (see Figure confusion matrix). On the diagonal the optimal condition would be to have values equal to 1, elsewhere the optimal condition would be to have values equal to zero. Looking at the confusion matrix for *classifier-EXT* (Figure 2), a confusion between, on the one side, syllables $B1/B2$, and, on the other side, between syllables $J1/J2$ can be seen on the diagonal in correspondence of these elements (sub-diagonal elements are darker than the others for these 4 syllables). This confusion can be explained with the fact that phrases formed by syllables $B1$ and phrases formed by syllables $B2$ on the one side, and phrases formed

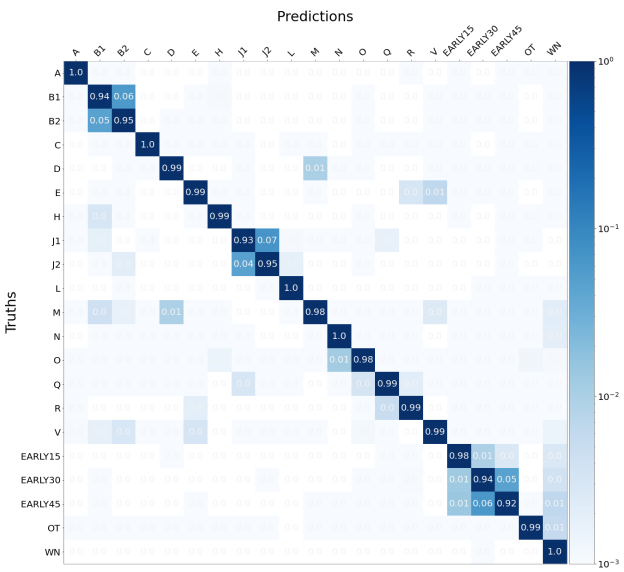


Fig. 2. **Confusion matrix *classifier-EXT***. Confusion matrix (i.e. the level of confidence of the classifier in making the correct assignment) relative to *classifier-EXT*. Each square represents the level of confidence of the classifier in making the correct assignment. The level of confidence is expressed on a logarithmic scale using shades of blue.

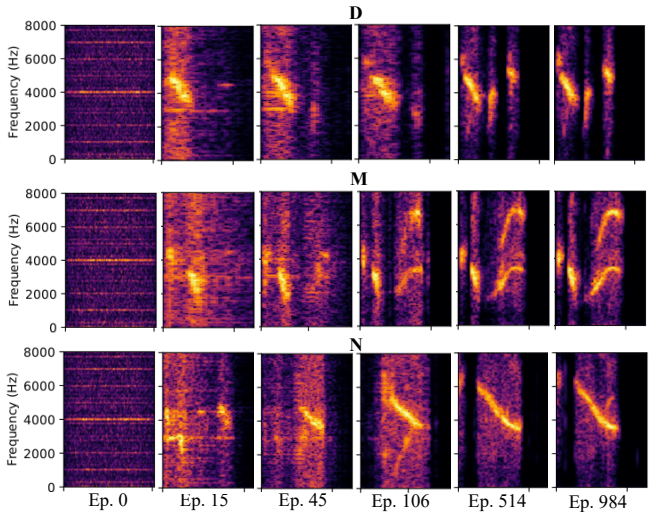


Fig. 3. **Genrations across time: syllable: example of three selected syllable across time**. The syllables has been first generated at epoch 984 and recognized using *classifier-EXT*. Then, the latent vector associated at each syllable has been used to generate the same syllable at epochs 0,15, 45, 106 and 514. From the top to the bottom: syllable *D*, syllable *M*, and syllable *N*. From the left to the right: at epoch 0 the generator produces a noisy sound, not resembling at a syllable; at epoch 15 the generator produces a sample that is coherent in duration but remain noisy and unclear; at epoch 45 the syllables is more distinguishable than at earlier epochs, but the remains noisy; as the training goes on, the generation resembles more and more real syllable *M* (epochs 106 and 514). The generation obtained at epoch 984 has a clear distinguishable shape, which can be compared with the one shown in Figure 1. Here, the generator obtained from the instance *Ex 6* in Figure 5 has been used to generate the syllables across time; latent space dimension $ld = 3$; *Ep.* stands for epoch.

by syllables $J1$ and phrases formed by syllables $J2$ on the other side, differ in repetition rate more than in frequency. The confusion matrix obtained from *classifier-REAL* can be found in the supplementary material (Figure 19).

The Inception Score (IS) obtained for the training dataset after using *classifier-REAL* to recognize the syllables is $IS_{real} = 15, 92$. The range of the IS for our dataset is $IS \in [1, 16]$.

B. Evaluation of the model

We used the classifier described in Section III-C1 to obtain a quantitative analysis. We saved the model every ~ 15 epochs until epoch ~ 1000. In order to determine a good epoch to generate samples resembling the real ones, we are interested in the performance of the generator across time. At the beginning of training (epoch 0 in Figure 3), the generator produces for all the classes a noisy output which does not resembles a real syllable. At epoch 15 the generator starts to produce a sound which is coherent in duration but remains noisy and unclear. The resemblance increases over time (at epoch 45, 106 and 514) and, finally, at epoch 984 the generator produces a syllable showing a shape comparable with the training data (Figure 1). An example for each class of the repertoire is provided in Figure 23 (supplementary material).

Quantitative evaluation

At each epoch, we generated $1k$ samples and we used *classifier-EXT* to calculate the probability distribution of the classes. Figure 4 shows the distribution obtained from the classifier at 4 example epochs: epochs 15, 106, 212, 318, 514 and 984. Each columns represents one of the 21 classes of the vocabulary: the 16 classes of the repertoire and 5 alternative unknown classes $x \in X$ (i.e., three classes of *EARLY* generations, the overtraining (*OT*) class, and the artificial white noise (*WN*)). These results have been obtained from an instance of training where the latent space dimension was fixed at $ld = 3$. It is possible to observe a decrease in the number of syllables belonging to an alternative class, whereas an increasing number of syllables characterize the classes of the repertoire.

The capacity of a 3-dimensional generator model to produce samples that are classified as belonging to the repertoire increases over time. After $\sim$ 200 epochs of training, the generator is able to cover all the syllables of the repertoire (Figure 5a). In parallel, the total variance (considering both the classes of the repertoire and the 5 alternative unknown classes) decreases over time (Figure 5c). That is, at early stages of training the majority of the samples generated are classified by *classifier-EXT* as elements of an alternative unknown class $x \in X$ (Figures 5d, 5e and 5f). Then, the number of samples recognized by *classifier-EXT* as belonging to a class of the repertoire increases leading to an increasing in the average number of syllables per class (Figure 5b). After epoch $\sim$ 600 the average number of syllables per class remains stable. At the same time, the number of syllables belonging to one of the alternative classes $x \in X$ (*EARLY15*, *EARLY30* and *EARLY45*) decreases over time (Figures 5d, 5e and 5f). Classes *EARLY15* and *EARLY30* decrease faster with respect to

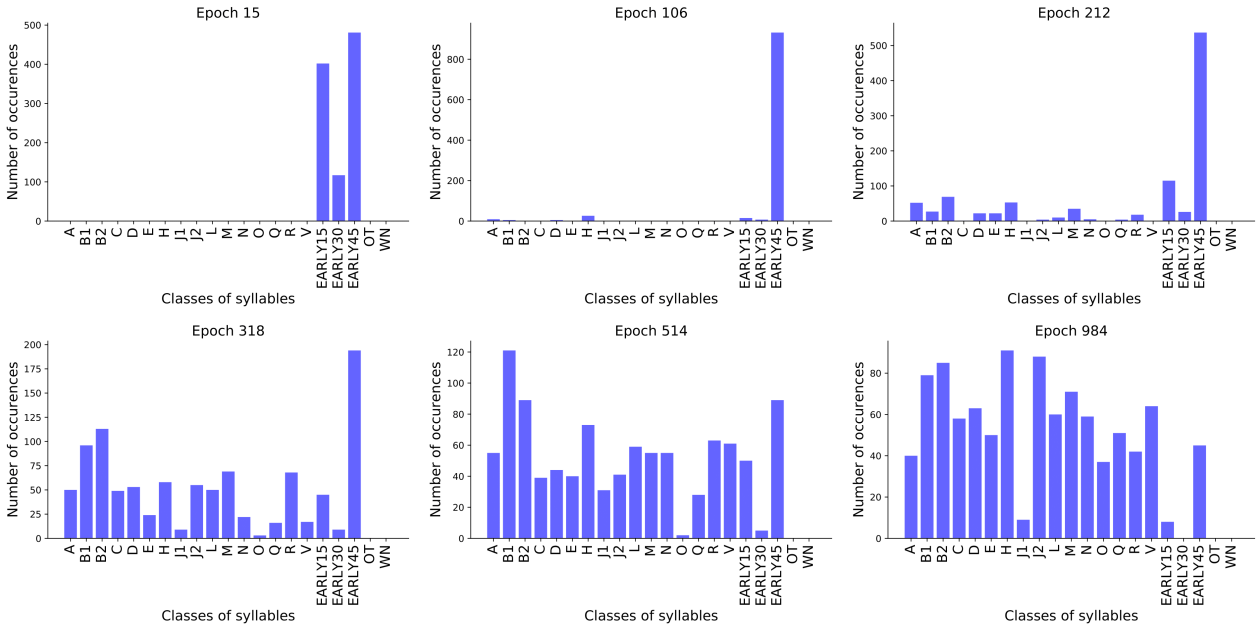


Fig. 4. **Distributionof the classes**. Distribution of $1k$ generated samples after 15, 106, 212, 318, 514, and 984 epochs of training. Each column represents a class of the vocabulary: a syllable from the repertoire, or an alternative unknown class $x \in X$ (*EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*). The number of syllables belonging to a class $x \in X$ decreases over time, whereas the average number of syllables belonging to the classes of the repertoire increases. The latent space dimension is $ld = 3$ and *classifier-EXT* has been used to classify the generated data.

EARLY45, which, eventually, never reaches a level comparable to the percentage found in the training data. Although this show that the generator produces better and better samples over time, further analysis are needed to understand if the quality of the samples keep increasing after epoch 600. The concept of overtraining arises comparing 10 different instances of training of 3-dimensional WaveGAN. The analysis of 10 instances of training of a 3-dimensional WaveGAN are shown in Figures 34 and 35 (supplementary material).

Finally, as a last quantitative measure of the generator performance, we computed the Inception Score (IS) across time. We generated $16k$ samples every ~ 100 epochs and we used *classifier-REAL* to classify them. Then, we computed the IS as in Equation 1 at each saved epoch. Figure 6 shows the IS across time with respect to IS_{train} (black line). The IS of *Ex 0* and *Ex 6* increases across time and, eventually, they reach a maximum value around 13 (for reference, the IS obtained from the training dataset is $IS_{train} = 15,92$); contrarily, the IS of *Ex 2* increases at the beginning but remains low during all the training. The IS of *Ex 0* suddenly breaks around epoch 800: this behavior confirms the results shown in Figure 34. Table I summarizes the IS of the training dataset and each of the selected instances: we considered the highest IS within the time range shown in Figure 6.

Qualitative evaluation

For qualitative evaluation, we focused on instance *Ex 6* of training. We chose to focus on this instance because it remains stable until epoch ~ 1000 (see Figure 5) and we obtained an increasing and high Inception Score (see Figure 6 and Table I). Indeed, *Ex 6* is also the example we used since the beginning of Section IV-B as a baseline example.

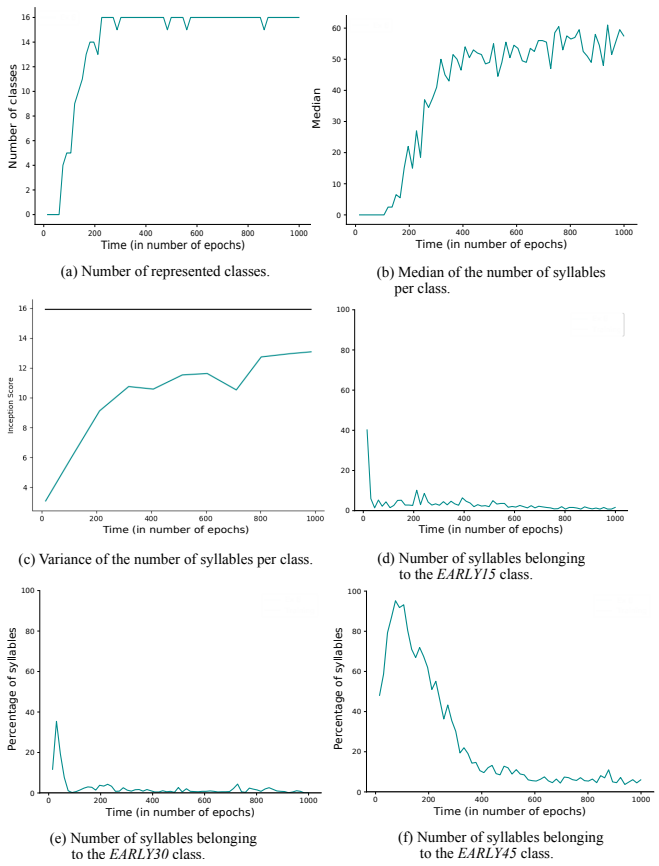
TABLE I
INCEPTION SCORE. IS OF THE TRAINING DATASET AND THE SELECTED INSTANCES: WE CONSIDERED THE HIGHEST IS WITHIN THE TIME RANGE SHOWN IN FIGURE 6. FOR OUR DATASET, $IS \in [1, 16]$.

Dataset	Epoch	IS
Training	-	15,92
Ex 0	800	13,51
Ex 1	211	8,91
Ex 2	515	5,65
Ex 3	512	12,29
Ex 4	605	11,89
Ex 5	408	11,08
Ex 6 (baseline)	984	13,07
Ex 7	988	13,61
Ex 8	802	12,66
Ex 9	605	12,19

The generator produces a higher number of good syllables over time and, at the end of the training, all the syllables can be produced by the generator. The mean spectrogram obtained from the generated data at epoch 984 can be compared with the one obtained from the training dataset (1b).

The UMAP representation of the generated data and the training data shows that (1) the generated data are grouped together as they were an additional cluster with respect to the ones obtained from the training data (Figures 8(a-d) and (2) the generated data spread over time, moving from being a cluster in itself (light blue points in Figures 8(a-d)) to taking a conformation compatible with the training data (brown points in Figures 8(a-d)). Further observations regarding the comparison between the UMAP representation of training and generated data are discussed in Figures25 and 26 in the supplementary material.

The UMAP representation of the generated data shows



(a) Number of represented classes.

(b) Median of the number of syllables per class.

(c) Variance of the number of syllables per class.

(d) Number of syllables belonging to the *EARLY15* class.

(e) Number of syllables belonging to the *EARLY30* class.

(f) Number of syllables belonging to the *EARLY45* class.

Fig. 5. **Analysis of a 3-dimensional generator using *classifier-EXT*.** Statistical analysis performed on the classifier distribution of one instance of the training. Panel (a) shows the number of classes represented by the generated data: that is, at each epoch, how many syllables of the repertoire are covered by the generator. The number of syllable reaches the maximum value 16 relatively quick and then stays high during all the training. Panel (b) shows the median of the number of elements per class: across time, with some instability, we can observe an increasing of the median across time. To compute the quantities in panels (a) and (b), alternative unknown classes $x \in X$ have not been taken into account. Panel (c) shows the evolution of the variance of how many syllable per class have been produced. Here, $x \in X$ classes are included. The variance starts at a high value when the majority of the samples produced are not classified as syllables of the repertoire, then it decreases when the generator becomes better at producing syllables. Panels (d-f) show the percentage of syllables that are classified as belonging to one of the alternative classes $x \in X$: from the left to the right, classes *EARLY15*, *EARLY30* and *EARLY45*.

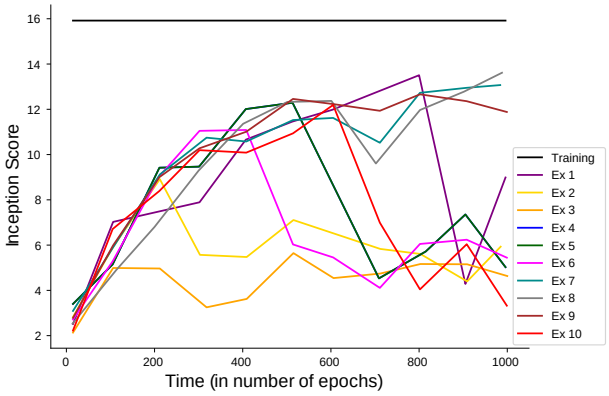


Fig. 6. **Inception Score across time.** IS relative to the generator of 10 instances of a 3-dimensional WaveGA, with respect to IS_{train} (black line) In particular, *Ex 6* (light blue line) is the instance used as baseline. IS for the some instances (e.g., *Ex 6* in light blue and *Ex 7* in gray) increases over time, whereas IS remains low over time in the case of instance *Ex 2* (orange line). Alternatively, it increases over time until it drops (*Ex 0* in purple and *Ex 9* in red) Here, for each instance we generated $16k$ samples every ~ 100 epochs and we used *classifier-REAL* to classify them.

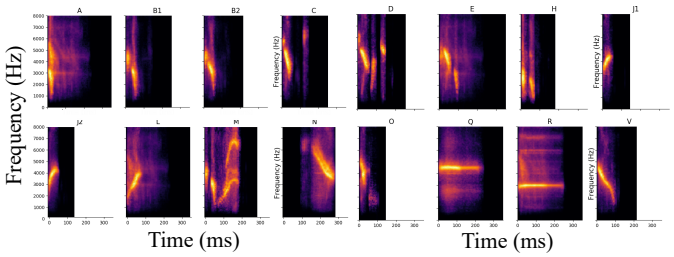


Fig. 7. **Mean spectrogram.** Mean spectrogram of $1k$ syllables generated at epoch 984. All the syllables are clear and distinguishable. They can be compared with the repertoire in Figure 1. Here, the training has been done using $ld = 3$, the generator obtained from the instance *Ex 6* in Figure 5 has been used to generate the syllables across time and *classifier-EXT* has been used to identify the generated syllables.

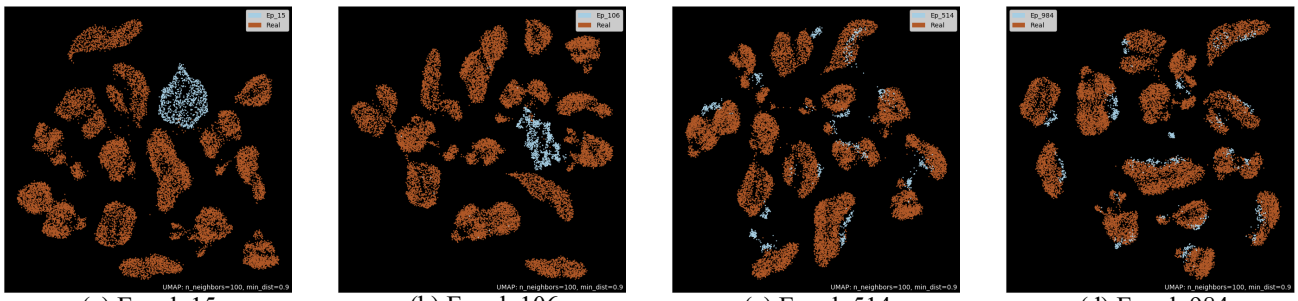
smoother transitions from one cluster to another (Figure 9). These transitions can be either represented by points belonging to one of the classes of the repertoire (e.g., between *J* and *C*) or to class *X* (e.g., between *N* and *Q*). This observation highlights an interesting perspective for which elements classified in class *X* can be seen as intermediate elements between two syllables. We will deal with this concept in Section IV-C.

Finally, human judgment confirms the goodness of the classifier and of the generated data. Each judge evaluated the same set of 200 samples generated at epoch 984 using the generator of instance *Ex 6*. The judges had the possibility to classify each of them as an element of one of the 16 classes of the repertoire or as an element of an alternative unknown class $x \in X$. The kappa coefficients κ_{Cohen} obtained across judges and with respect to *classifier-EXT* are comparable: $\kappa_{Cohen}(Judge1, Judge2) = 0,74$, $\kappa_{Cohen}(Judge2, \textit{classifier-EXT}) = 0,79$ and $\kappa_{Cohen}(Judge1, \textit{classifier-EXT}) = 0,73$ (these values are collected in Table III in the supplementary material).

For further observations about the stability of the training, Appendix IV-E shows how, combining the UMAP representation and the mean spectrogram, it is possible to compare three consecutive epochs of training to check the stability of the generator.

C. Latent space exploration

To explore the structure of the latent space, we used *Ex 6* at epoch 984. We compared the UMAP representation obtained from $16k$ generations after training (Figure 9) with the corresponding latent vectors (Figure 10). The latent space is dense (Figure 10a), i.e. a high number of latent vectors originate a syllable that *classifier-EXT* recognizes as an element of the repertoire, and only a minority of elements is classified as belonging to class *X*. The latent space structure is comparable with the structure observed for the spectrogram representation of the generations. Clusters that are close in the data (Figure 9) are close in the latent space (Figure 9b): for instance, the



(a) Epoch 15. (b) Epoch 106. (c) Epoch 514. (d) Epoch 984.

Fig. 8. **Syllable space representation across time.** Syllable space representation obtained from the training dataset ($16k$ syllables) and $1k$ syllables generated at epochs 15 (a), 106 (b), 514 (c) and 984 (d) using UMAP [31]. Brown points represent the training data, while light blue points the generated data. These four figures are different because the analyzed dataset differs for the $1k$ generations specific of each epoch. Here, the training has been done using $ld = 3$, the generator of instance *Ex 6* has been used to generate the syllables.

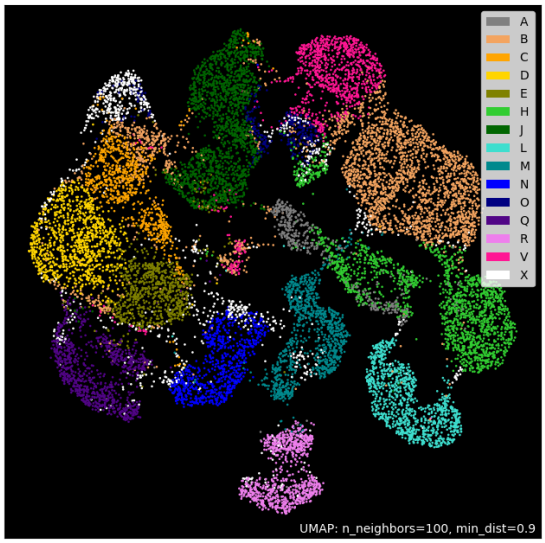


Fig. 9. **UMAP representation of the generated data.** UMAP representation of $16k$ generated syllables. Each cluster/color correspond to one class of the repertoire and class *X* (in white) represents the cumulative class of the alternative unknown classes (in this case, *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*). Here, the training has been done using $ld = 3$, the generator obtained from the instance *Ex 6* in Figure 5 has been used to generate the syllables at epoch 984 and *classifier-EXT* has been used to identify the generated syllables.

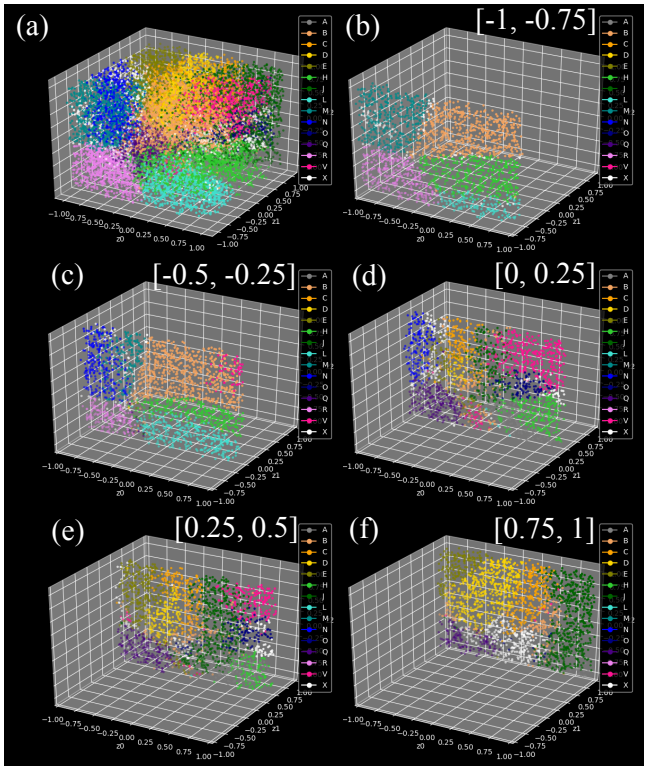


Fig. 10. **Structure of the latent space**. We selected $16k$ random vectors $z \sim R^3([-1, 1])$ from a 3-dimensional latent space, and we generated $16k$ syllables. Then, we used *classifier-EXT* to classify each new generation. Each point in panels (a-f) corresponds to a 3-dimensional latent vector. Each color corresponds to a class of the vocabulary (i.e., a class of the repertoire or an alternative class). As for the UMAP representation in Figure 9, the alternative classes have been represented a cumulative class *X*. Panel (a) shows a global representation containing all the $16k$ latent vectors, giving an idea of the density of the latent space. Panels (b-f) represent five slices from the whole space. Points close in the latent space are also close in the spectrogram representation (Figure 9). The difference between the structure of the clusters can be observed: some of them are more sparse than others (e.g., the dark blue cluster representing syllable *O*, where a lot of white points - i.e., elements belonging to class *X* - mix up with syllables classified as belonging to *O*). The syllables have been produced by the 3-dimensional generator obtained from instance *Ex 6*.

clusters relative to syllables *B* (cream cluster), *H* (light green cluster), *L* (light blue cluster), *R* (light pink cluster) and *J* (turquoise cluster). Moreover, some clusters (e.g., syllable *O*) show an higher sparseness than others (dark blue cluster in Figures 10e and 10f).

To explore the continuity of the latent space we used the generator obtained from one instance of training of a 3-dimensional WaveGAN. We used two different strategy to (1) explore the latent space in all the direction starting from one point (one component variation described in Section III-C2) and (2) observe the transition from one syllable to another (three components variation described in Section III-C2). First, we selected a random latent vector $z \sim R^3([-1, 1])$ to create a baseline syllable, then continuously move from one representation to another by changing one dimension of the latent space

by a fix variation step. The syllables highlighted by the red squares in Figure 11 (i.e., first *C* and *J2*, then *V* and *V*) are an example of non-smooth transitions. For these particular transitions, we considered the two consecutive syllables obtained from the first variation step (i.e., two consecutive syllables contained in two consecutive red squares) and we applied a smaller variation step. The bottom-left panel of Figure 11 shows the latter operation applied to the first non-smooth transition in the first component and highlights a new non-smooth transition that needs to be investigated. Using the same procedure, we applied a smaller variation step and reached a point at which we cannot see any more clear evidence of non-smooth transitions (bottom-right panel of Figure 11). The investigation of the second non-smooth transition highlighted in the upper panel of Figure 11 and additional examples from the the second and the third components can be found in the supplementary material.

A better visualization of such smooth transition can be observed in the UMAP representation shown in Figure 12a. The exploration obtained varying, one by one, the first component of a random latent vector $z \sim R^3([-1, 1])$ by a step equal to $v_{step} = 0.001$ shows smooth passages between one syllable to another. It is not surprising that a difficulty in the classification arises between syllable *V* (magenta cluster) and syllable *C* (orange cluster): indeed, syllable *V* is a syllable that shares its content with other syllables of the repertoire, causing uncertainty and errors for the classifier. The UMAP representation of the training dataset (Figure 1a) and then the UMAP representation of the generated data (Figure 9) already show multiple cluster locations for syllable *V*. Additional figures and comments about the certainty of the classifier can be found in the supplementary material (Figures 21, 22 and 32).

We used $N_{steps} = 1000$ and we computed the three components transition from syllable *M* to syllable *D*. These two syllables are not adjacent in the UMAP representation of the generated data (see Figure 9). The variational data cross the generated data (left panel of Figure 10b) giving rise to a smooth change of syllable class (right panel of Figure 10b). Interestingly, the transition between syllable *M* and syllable *D* (Figure 10b, right panel) shows that the intermediate syllables between two classes are sometimes recognized as class *X* (white points between syllable *M* (turquoise cluster) and syllable *N* (blue cluster). More examples of transition between syllables can be found in Figure 33 (supplementary material).

The latent space exploration, combined with the UMAP representation we obtained for the generated data and shown in Figure 9, highlight the fact that the generator is not only producing samples from the training dataset but also other samples. This allows to move realistically between two syllables, as shown in Figure 10a.

D. Latent space dimension

Until now, we have shown results obtained from a 3-dimensional WaveGAN. Indeed, a 3-dimensional WaveGAN works nicely and as good as higher dimensional WaveGANs and, at the same time, lower dimensional WaveGANs do not show good performances or stability (Figure 13). Although all conditions allows the generator to reach the ability of producing all the type of syllables, 1-dimensional and 2-dimensional WaveGANs converge later with respect to higher dimensional WaveGANs (Figure 13a). At the same time, 1-dimensional and 2-dimensional WaveGANs show a slower increase in the average number of syllables belonging to each class of the repertoire (Figure 13b) and a slower decrease in the variance of the number of syllables per class. Nevertheless, as already mentioned in Section IV-B, towards the end of the training a generalized instability could appear (what we refer to as overtraining), without distinction of latent space dimension (e.g., overtraining happens for $ld = 4, 5, 6$ - respectively, the blue, the green and the pink line). This can be seen in a decrease in the median evolution (Figure 13b) and in the sudden drop in the inception score (Figure 13c).

A qualitative measure of the generated syllables obtained from WaveGANs of different dimensions (e.g., for different conditions of the latent space dimension) can be found in Figure 37 (supplementary material).

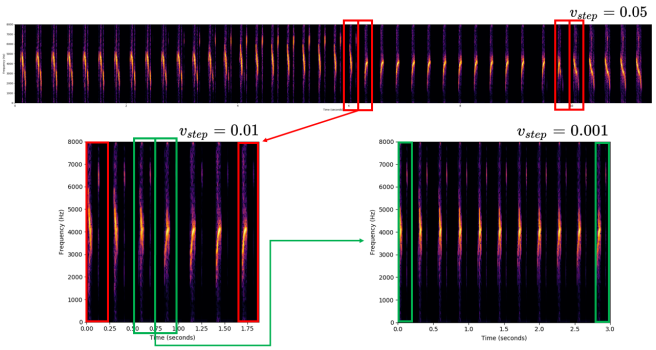
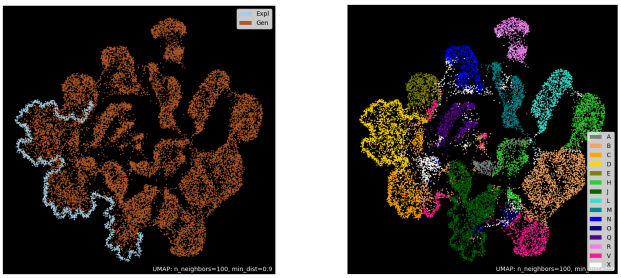


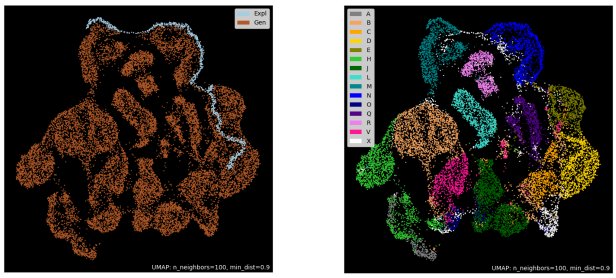
Fig. 11. **Exploration of the latent space: one component variation. First component.**. We selected a random latent vector $z \sim R^3([-1, 1])$ to create a baseline syllable. Then, we moved one by one the components of the vector by a variation step equal to $v_{step} = 0.05$. We observed all the syllables produced to look at how they evolve and if there are non-smooth transitions. The upper panel shows the exploration of the first component of the latent vector obtained with $v_{step} = 0.05$. The syllables highlighted by the red squares (i.e., first *C* and *J2*, then *V* and *V*) are an example of non-smooth transitions. For these particular transitions, we considered the two consecutive syllables obtained from the first variation step (i.e., two consecutive syllables contained in two consecutive red squares) and we applied a variation step of $v_{step} = 0.01$ to the first component of the latent vector to generate intermediate syllables. The bottom-left panel shows the latter operation applied to the first non-smooth transition in the first component and highlights a new non-smooth transition that needs to be investigated. To do so, as shown in the bottom-right panel, we used a variation step equal to $v_{step} = 0.001$ to generate intermediate syllables. We have reached a point at which we cannot see any more clear evidence of non-smooth transitions. The syllables have been produced by the 3-dimensional generator obtained from instance *Ex 6* and the name of the syllable for this analysis has been provided by *classifier-EXT*.

E. Training dataset dimension

A larger dataset allows WaveGAN to converge faster and better (Figure 14). At the beginning of the training the generator was only able to produce a limited number of syllable types, whereas it becomes able to reproduce all of them after some epochs of training (Figure 14a). Only the smaller dataset (green line) shows a slower convergence to 16. Likewise,



(a) Variation in the first component



(b) Transition from M to D

Fig. 12. **Latent vector exploration.** Panel (a) represents the first component variation. A latent vector has been randomly selected and a step $v_{step} = 0.001$ has been applied to its component, one by one, in oder to generate $1k$ syllables. Panel (b) represents the transition between *M* (turquoise cluster) and syllable *D* (yellow cluster). The transition between one syllable to the other has been done generating intermediate syllables as described in Equation 2 (Section III). The left panel show the variational data (generated at each step) (light blue points) and $16k$ generated data from the same model at the same epoch (brown points). Each cluster/color in the right panels corresponds to one class of the repertoire and class *X* (in white) represents the cumulative class of the alternative unknown classes (in this case, *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*). Here, the training has been done using $ld = 3$. The generator obtained from instance *Ex 6* of the training has been used *classifier-EXT* has been used to identify the generated syllables from the generator of instance *Ex 6*.

the average number of syllables generated per class across time increases more when WaveGAN is trained with a larger dataset (Figure 14b). Finally, the inception score increases over time no matter which is the size of the training dataset, but it reaches a higher values for a bigger dataset (blue line in Figure 14c).

V. Discussion

A. Summary of the main results

In this paper we tested the ability of WaveGAN [15] to produce canary syllables when the latent space dimension is low. Originally, Donahue et al. [15] trained WaveGAN on a set of wild recordings from different bird species: the generated samples were noisy due to the variability of the dataset without enough recordings for each specie. By introducing a dataset of recordings from a single adult canary, we simplified the complexity of the training dataset and obtained qualitatively better results. The fact that we deal with single syllables simplifies the analysis of the performance of the generator and the exploration of the latent space. Moreover, WaveGAN has been trained with a 100-dimensional latent space.

We showed that WaveGAN can produce good syllables even if the latent space dimension is reduced to $ld = 3$. The generator can produce sounds with similar acoustic properties as the canary syllables used to train it, with a diversity of produced sounds that covers the training dataset and interpolate in between training data. We relied on a RNN-based classifier to recognize the generated syllables and evaluate them quantitatively and qualitatively. On the one hand, we looked at the statistical properties of a set of generated data (e.g., average and median number of syllables produced per class, variance, inception score). On the other hand, we used the mean spectrogram and UMAP [31] representations (1) to compare generated data with training data and (2) to study the stability of training. We explored the latent space to highlight the continuity in the representation of acoustic features of the sounds: we first explored how small changes in the latent space influence the generations (see Figure 11); then, we interpolated between two well-defined syllables (see Figure 12a). We concluded that a low-dimensional GAN (dim=3) trained with a reasonnable dataset (16 syllables x 1000 iterations) is able to produce good quality canary syllables, to generalize and to interpolate between syllables.

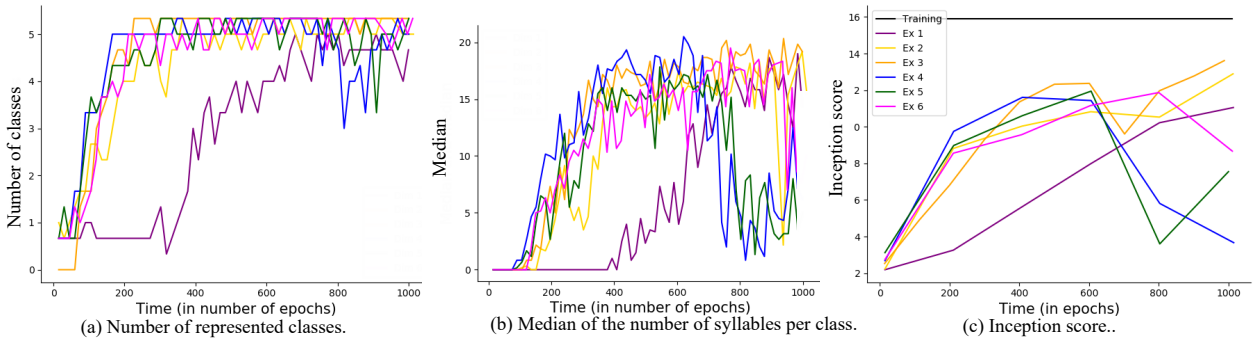
B. Limitations

The limitations of this work are related to the availability of the data, and to the intristic characteristics of the tools we chose to use. On the one end, it is not easy to obtain a well segmented and labeled canary, or songbird, dataset. On the other hand, there is still a poor understanding of the mechanisms involved in the training of GANs. First, a limitation of WaveGAN [15], and GANs in general, is the absence of a stopping criteria for the training: this introduces an incertitude about how to evaluate the learning and its stability (i.e., in number of epochs rather than depending on the loss function value). Moreover, the hyperparameter sensibility and the limitations related to overfitting may introduce some complications in the exploration of the GANs performance.

C. Evaluation of the performances

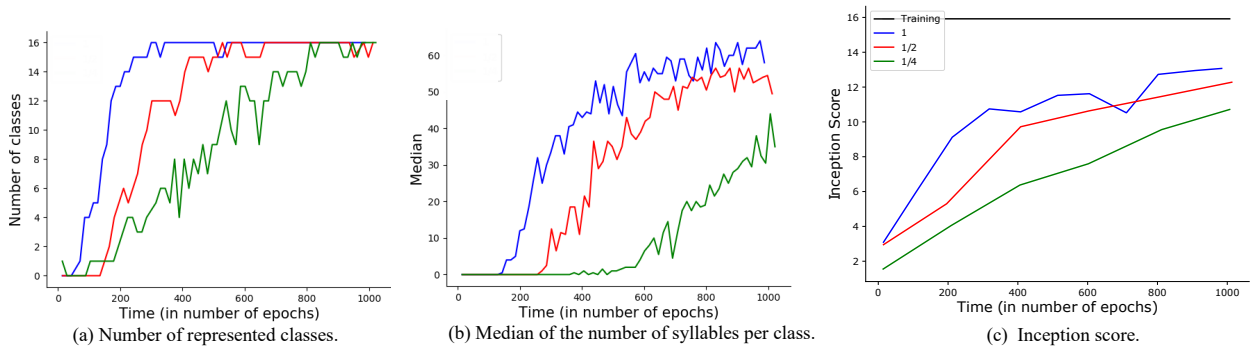
There is currently no consensus on how to evaluate generative models, and especially GANs [23], [29]. However, generative models are generally evaluated using both a quantitative and a qualitative measure of the quality and diversity of their output. A quantitative evaluation aims to determine whether or not the generator is producing only examples on which the classifier was trained. A qualitative measure is necessary to truly understand the quality of the produced samples and their comparability to real recordings in term of whether or not they are comprehensible to an external judge. At the same time, the drawbacks of a qualitative measure is that it presupposes the interpretability of the data by humans, which is not always possible [29]. Moreover, it could be biased, expensive, not efficient in detecting overfitting.

1) Quantitative evalutation: Inception Score (IS) [13] is a broadly used quantitative measure which gives an idea about whether or not the generator model is able to produce a wide enough variability in the samples. Nevertheless, IS is not able to detect if the generator is producing only examples that belong to the training dataset or if the generator enter mode



(a) Number of represented classes. (b) Median of the number of syllables per class. (c) Inception score..

Fig. 13. **Comparison between different latent space dimensions**. Each line represents one instance of training at a particular latent space dimension. Panel (a) shows how many syllables of the repertoire are covered by the generator across time. Panel (b) shows how many syllable per class have been generated in average. The dark lines show the evolution of the mean, whereas the light lines shows the evolution of the median. To build these two panels (a) and (b), $1k$ generations have been generated every 15 epochs, *classifier-EXT* has been used to provide the classification, and only the repertoire's classes have been taken into account when plotting. Panel (c) shows the evolution of the inception score over time. To build this panel $16k$ syllables have been generated every 200, and *classifier-REAL* has been used to provide the classification. A 3-dimensional WaveGAN (orange line) reaches convergence as good as higher-dimensional WaveGANs (blue, green and magenta lines) and better than lower dimensional WaveGANs (purple and yellow lines). We varied the latent space dimension as $ld = 1, 2, 3, 4, 5, 6$ and we kept the training dataset fixed. A complete version of this figure can be found in the supplementary material (Figure 36).



(a) Number of represented classes. (b) Median of the number of syllables per class. (c) Inception score.

Fig. 14. **Comparison between datasets of a different size**. We varied the dataset size as $d = 23456, 3600, 1600$ and we kept fix the latent space dimension at $ld = 3$. Each line in this figure represents one instance of training at a particular dataset size condition. Panel (a) shows how many syllables of the repertoire are covered by the generator across time. Panel (b) shows how many syllable per class have been generated in average. The dark lines show the evolution of the mean, whereas the light lines shows the evolution of the median. To build these two panels (a) and (b), $1k$ generations have been generated every 15 epochs, *classifier-EXT* has been used to provide the classification, and only the repertoire's classes have been taken into account when plotting. Panel (c) shows the evolution of the inception score over time. To build this panel $16k$ syllables have been generated every 200, and *classifier-REAL* has been used to provide the classification. A dataset of bigger size (blue line) allows better and faster convergence than having a dataset of lower sizes (red and green lines). A complete version of this figure can be found in the supplementary material (Figure 38).

collapse (i.e., it produces always the same class) [39]. Thus, IS should not be used as a holistic method to evaluate the performance of a model and must be combined with another evaluation method [40]. Instead, an additional quantitative measure could be used to have a better understanding of the quality of the training. For instance, Donahue et al. [15] measured the Euclidian distance in the space of log-Mel spectrograms (1) within the training and the generated data (to measure the intra-dataset diversity with respect to the training data), and (2) between the training and the generated data (to show the ability of WaveGAN of producing sounds not belonging to the training dataset). During the training, the median of the number of syllables produced per class continuously increase over time (see Figure 34). A similar behavior can be seen for the IS (see Figure 6). Eventually one can observe a drop if overtraining begins: we call overtraining the fact that, at a certain epoch, there is a drop in the IS (see Figure 6) or in the statistical properties of the classifier distribution (see Figure 34b). However, the concept of overtraining remains unclear. We tried to characterize it using a specific class (*OT*) which arises at advanced stages of training, but never becomes the most represented class (see Figure 35). Instead, after the training drops, there is an exponential increase in the number of element assigned to early classes (see Figure 34(d-f)).

2) Qualitative evalutation: A qualitative evaluation based on human judgment has been proposed by Salimans et al. [13] and Denton et al. [41] to evaluate GANs trained on MNIST or CIFAR-10 (image datasets). Similarly, Donahue et al. [15] used human judgment to evaluate WaveGAN performance when training on SC09 (speech dataset). We rely first on the mean spectrogram obtained from the generated data over time (see Figure 7 and Figure 24) to observe at once how the average quality of syllables increases over time, becoming more and more comparable with the repertoire (see Figure 1). Then,

we rely on the UMAP [31] representation of the spectrograms to compare the generated data with the training data. On the one hand, the generated data belong to the same cluster as the training data (see Figure 8) if represented together. On the other hand, a continuity across clusters arises when observing the generated data alone (see Figure 9). In the case of data not familiar to humans (such as canary syllables), one could design a behavioral protocol to test the accuracy by measuring how canaries perceive the generated data. Although such an experiment could be very interesting, it is complicated to settle and require an expertise outside of the machine learning domain.

D. Structure of the latent space

The exploration of the latent space allows to study whether or not the generator is able to produce a continuous space, and interpolate between its element [11]. Moreover, it allows to compare the structure of the sound output space (spectrograms) and the latent space: a desirable property is that clusters that are close by in the sound space are close by in the latent space [10]. The exploration of the latent space by interpolating between different latent space elements has been proposed by Drysdale et al. [42] after training WaveGAN on a drums dataset, and by Blaauw and Bonada [43] after training a Variational Autoencoder (VAE) on a speech corpus. Alternatively, Hsu et al. [44] train a VAE on a speech corpus and decompose the latent space into orthogonal latent attribute representations and explore how speech attribution changes (e.g., they move from a female speaker to a male speaker).

In our study, we analyzed (1) how a small change in the latent space affects the generated syllable, (2) the transition between two different generated syllables, and (3) whether or not clusters that are close by in the spectrogram space (obtained using UMAP) are close by in the latent space. The training dataset generates well separated clusters in the UMAP representation, as shown in Figure 1a, whereas the clusters obtained from generated data appears more continuous with transition "paths" between clusters (see Figure 9). By interpolating between two syllables in the latent space using small steps, it is possible to move continuously in the UMAP representation of the spectrograms (see Figure 12a). Alternatively, if the applied step in the latent space is too big there are non-smooth transitions between syllables (see Figure 11). To our knowledge, there is no other GAN performing exploration of one latent dimension at a time (see Figures 11 and 12a). This is possible because we have a low-dimensional GAN latent space while previous works have high-dimensional ones.

E. Perspectives

As future work to this study, several perspectives could be considered. In the next paragraphs, we summarize how the WaveGAN generator could help the understanding of syllable generation in songbirds when using a different training dataset, or when integrating it in a more complex model (e.g., a vocal learning model).

1) Different training dataset: One could explore the possibility of enlarging the training dataset by including (1) more classes of syllables, (2) more birds (in terms of number), (3) recordings from different species or (4) recordings from juveniles. In principle, the addition of more syllables or recordings from different canaries would introduce complexity in the training dataset and would increase the computational time (i.e., the number of epochs needed to obtain a generator able to output good syllables). On the contrary, the attempt of using the model on other species is not trivial. For instance, although zebra finches represent an important model for vocal learning, they produce syllables with harmonic components. Thus, it is more difficult to identify/evaluate for a human observer compared to canary syllables. Training WaveGAN on both juveniles and adult data is an interesting and rather straightforward extension of this work. For instance, it can be useful to design a motor control function that could produce any possible canary sounds, from juveniles to adults. This would be useful in computational experiments in order to model the sound productions at different stages of learning. It could also be useful for behavioral experiments with birds. A similar experiment has been proposed by Sainburg et al. [38]: generated samples from a Variational Autoencoder (VAE) has been played to a group of European starlings in a decision making experiment. The birds can learn the task with a high proficiency, and the neural responses obtained from electrophysiology vary smoothly when small variations are applied to the stimulus.

2) Vocal learning model: Our results support the use of generative models as motor function in vocal learning models, as an alternative to other sound synthesis. Indeed, motor control is most often modeled using Ordinary Differential Equations (ODEs) representing the sound generation organ in vocal learning models [45]–[48]. Such models are usually based on the anatomical structure of the vocal tract and the respiratory apparatus, and, especially for humans, can include a large number of parameters. However, learning to control a high-dimensional motor apparatus in current theoretical frameworks for sensorimotor learning remains challenging [8], [19], [49] and often leads to unrealistic convergence time [9]. Moreover, a lower dimensional motor control reduces the number of parameters that need to be learned, thus simplifying the learning (w.r.t an equivalent topology of latent space). Reducing the number of dimensions controlled by the learning algorithm therefore appears as a necessary first step to design vocal learning models with more biologically plausible learning rules. Consequently, we believe in the possibility that generative models, and in particular the generator of WaveGAN, can be used as motor control in vocal learning models. Biologically, they would represent a premotor layer rather than the control parameters of a vocal organ. Such premotor "module" simplifies the dimensionality of the learning problem and can be seen as an equivalent of embodiment: appropriate body features can facilitates the motor control algorithms [50].

3) Generation of canary songs: A generator model able to produce sequences of syllables would be difficult to evaluate: indeed, it would be difficult to perform the qualitative analysis

we propose here (e.g., the mean spectrograms or the transition from one syllable to another in the latent space). Alternatively, a model able to generate single syllables, as the one proposed in this work, may be an adequate tool to generate full canary songs. Such model would require (1) an estimation of the distribution of the delay within syllables of the same class and (2) a probabilistic model estimating (i) how many time a syllable is repeated and (ii) the upcoming syllable in the phrase. Indeed, canary songs are composed of sequences of phrases, which are themselves repetitions of the same syllable with smooth transitions. The number of syllable repetitions in one phrase is variable. Thus, a GAN trained to produce bouts of a few seconds would not be able to reproduce the variability of canary song with much flexibility. On the contrary, a GAN like the one proposed, which is able to produce smooth controllable transitions between syllables, is likely to be a good tool to generate full canary songs.

Finally, unrelated to birdsong domain, such low-dimensional GAN and the diverse explorations it enables can help in a general fashion for generative model future works.

ACKNOWLEDGMENT

We would like to thank Catherine Del Negro, Aurore Cazala and Juliette Giraudon for the recording and transcription of the canary data. We also thank Inria for the CORDI-S PhD fellowship grant and the LabEx BRAIN for the PhD extension grant. Experiments presented in this paper were carried out using the PlaFRIM experimental testbed, supported by Inria, CNRS (LABRI and IMB), Université de Bordeaux, Bordeaux INP and Conseil Régional d'Aquitaine (see https://www.plafrim.fr/).

REFERENCES

[1] J. Hillenbrand, L. A. Getty, M. J. Clark, and K. Wheeler, "Acoustic characteristics of american english vowels," *The Journal of the Acoustical society of America*, vol. 97, no. 5, pp. 3099–3111, 1995.

[2] J. L. Miller and A. M. Liberman, "Some effects of later-occurring information on the perception of stop consonant and semivowel," *Perception & Psychophysics*, vol. 25, no. 6, pp. 457–465, 1979.

[3] P. Kuhl, "Early language acquisition: cracking the speech code," *Nature reviews neuroscience*, vol. 5, no. 11, p. 831, 2004.

[4] M. Chakraborty and E. Jarvis, "Brain evolution by brain pathway duplication," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 370, no. 1684, p. 20150056, 2015.

[5] A. Doupe and P. Kuhl, "Birdsong and human speech: common themes and mechanisms," *Annual review of neuroscience*, vol. 22, no. 1, pp. 567–631, 1999.

[6] J. E. Markowitz, E. Ivie, L. Kligler, and T. J. Gardner, "Long-range order in canary song," *PLoS Comput Biol*, vol. 9, no. 5, p. e1003052, 2013.

[7] A. K. Dhawale, M. A. Smith, and B. P. Ölveczky, "The role of variability in motor learning," *Annual review of neuroscience*, vol. 40, pp. 479–498, 2017.

[8] R. Parr and S. Russell, "Reinforcement learning with hierarchies of machines," in *Advances in neural information processing systems*, 1998, pp. 1043–1049.

[9] S. Pagliarini, X. Hinaut, and A. Leblois, "A bio-inspired model towards vocal gesture learning in songbird," in *2018 Joint IEEE 8th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*. IEEE, 2018, pp. 269–274.

[10] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A hierarchical latent vector model for learning long-term structure in music," *arXiv preprint arXiv:1803.05428*, 2018.

[11] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[12] C. Y. Lee, A. Toffy, G. J. Jung, and W.-J. Han, "Conditional wavegan," *arXiv preprint arXiv:1809.10636*, 2018.

[13] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in neural information processing systems*, 2016, pp. 2234–2242.

[14] H.-W. Dong, W.-Y. Hsiao, L.-C. Yang, and Y.-H. Yang, "Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[15] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," *arXiv preprint arXiv:1802.04208*, 2018.

[16] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[17] G. Giraudon, N. Trouvain, A. Cazala, C. Del Negro, and X. Hinaut, "Labeled songs of domestic canary m1-2016-spring (serinus canaria) (version 0.0.1) [data set]." *Zenodo*, vol. http://doi.org/10.5281/zenodo.4736597.

[18] F. Benureau, "Self exploration of sensorimotor spaces in robots," pp. English. NNT : 2015BORD0072 . tel–01 251 324v2, 2015.

[19] M. Rolf, J. Steil, and M. Gienger, "Goal babbling permits direct learning of inverse kinematics," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 3, pp. 216–229, 2010.

[20] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[21] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[23] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," *arXiv preprint arXiv:1701.00160*, 2016.

[24] D. Berthelot, T. Schumm, and L. Metz, "Began: Boundary equilibrium generative adversarial networks," *arXiv preprint arXiv:1703.10717*, 2017.

[25] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," *arXiv preprint arXiv:1609.03126*, 2016.

[26] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.

[27] H. Petzka, A. Fischer, and D. Lukovnicov, "On the regularization of wasserstein gans," *arXiv preprint arXiv:1709.08894*, 2017.

[28] C. Raffel, "Lakh midi dataset (lmd)," *http://colinraffel.com/projects/lmd*, 2016.

[29] A. Borji, "Pros and cons of gan evaluation measures," *Computer Vision and Image Understanding*, vol. 179, pp. 41–65, 2019.

[30] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[31] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[32] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and psychological measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[33] R. Artstein and M. Poesio, "Inter-coder agreement for computational linguistics," *Computational Linguistics*, vol. 34, no. 4, pp. 555–596, 2008.

[34] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, vol. 148, no. 34, p. 13, 2001.

[35] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[36] N. Trouvain and X. Hinaut, "Canary song decoder: Transduction and implicit segmentation with ESNs and LTSMs," *HAL preprint hal-03203374*, 2021.

[37] N. Trouvain, L. Pedrelli, T. T. Dinh, and X. Hinaut, "Reservoirpy: an efficient and user-friendly library to design echo state networks," in *International Conference on Artificial Neural Networks*. Springer, 2020, pp. 494–505.

[38] T. Sainburg, M. Thielk, and T. Q. Gentner, "Latent space visualization, characterization, and generation of diverse vocal communication signals," *bioRxiv*, p. 870311, 2019.

[39] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," *arXiv preprint arXiv:2001.06937*, 2020.

[40] S. Barratt and R. Sharma, "A note on the inception score," *arXiv preprint arXiv:1801.01973*, 2018.

[41] E. L. Denton, S. Chintala, R. Fergus *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, 2015, pp. 1486–1494.

[42] J. Drysdale, M. Tomczak, and J. Hockman, "Adversarial synthesis of drum sounds," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2020.

[43] M. Blaauw and J. Bonada, "Modeling and transforming speech using variational autoencoders," *Morgan N, editor. Interspeech 2016; 2016 Sep 8-12; San Francisco, CA.[place unknown]: ISCA; 2016. p. 1770-4.*, 2016.

[44] W.-N. Hsu, Y. Zhang, and J. Glass, "Learning latent representations for speech generation and transformation," *arXiv preprint arXiv:1704.04222*, 2017.

[45] A. Amador, Y. Perl, G. Mindlin, and D. Margoliash, "Elemental gesture dynamics are encoded by song premotor cortical neurons," *Nature*, vol. 495, no. 7439, p. 59, 2013.

[46] T. Gardner, G. Cecchi, M. Magnasco, R. Laje, and G. B. Mindlin, "Simple motor gestures for birdsongs," *Physical review letters*, vol. 87, no. 20, p. 208101, 2001.

[47] G. Westerman and E. R. Miranda, "Modelling the development of mirror neurons for auditory-motor integration," *Journal of new music research*, vol. 31, no. 4, pp. 367–375, 2002.

[48] S. Maeda, "Compensatory articulation in speech: analysis of x-ray data with an articulatory model," in *First European Conference on Speech Communication and Technology*, 1989.

[49] R. Reinhart, *Reservoir computing with output feedback*. PhD Thesis. Bielefeld University, Germany, 2011.

[50] R. Pfeifer and J. Bongard, *How the body shapes the way we think: a new view of intelligence*. MIT press, 2006.

SUPPLEMENTARY MATERIAL

I. AUTOMATIC SELECTION

A. Pre-processing

As introduced in Section III-A, to select the syllables we used three parameters: amplitude threshold, minimal duration of the syllable and duration of the gap between two consecutive syllables. Figure 15 shows a summary of the selection procedure for syllable $J2$. The upper panel shows the downsampled phrase. The lower panel highlights each syllable of the phrase: for each syllable, the green dashed line represents the onset (i.e., the beginning of the syllable), and the blue dashed line represents the offset (i.e., the end of the syllable).

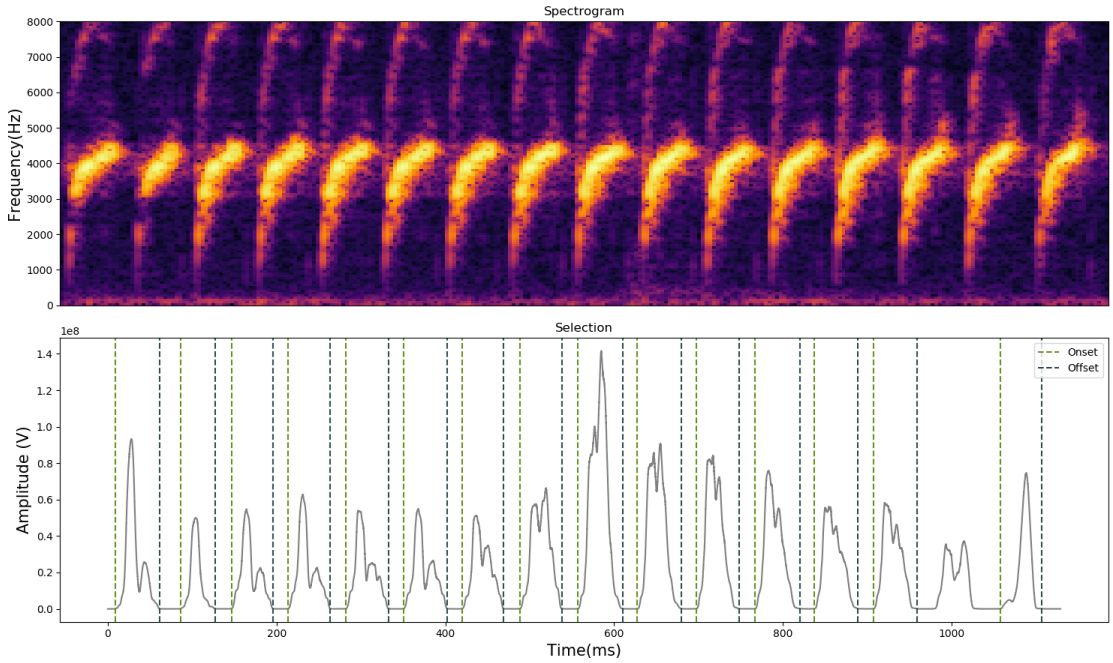


Fig. 15. **Selection of syllable $J2$.** The upper panel shows the spectrogram of an example phrase of syllable $J2$. The lower panel shows the selected syllables: the green vertical lines represent the onset of each syllable, the blue vertical lines represent the offset of each syllable. We used onset and offset to determine where each syllable begins and ends.

B. Semi-automatic error detection

We manually check the syllables after the automatic selection. In this section we explain which errors could arise from the automatic selection and how we solved them.

First, our algorithm could have failed in cutting a recording because of the presence of a too short gap between syllables. In this case we obtained samples containing more than one syllables. Alternatively, some samples could be too short, which means they contain only a part of the syllable. Finally, a bad initial classification (e.g. a phrase of class A was wrongly classified as class M) could lead either to well-selected syllables belonging to the wrong class, or, again, to a not effective cut. Applying a semi-automatic procedure for syllable selection, including errors, we were able to select 78827 syllables from the 16 classes. Some syllables (such as syllables belonging to class O, N, or C) are more difficult to select. Often the automatic selection based on amplitude threshold, duration of the syllable and of the gap, fails to select completely the syllable. That is, the beginning or the end of the syllable is systematically not recognized. In this case, we tried to correct manually the selection by adding a certain amount of samples after (or before) the automatic selection: in this way, more syllables can be correctly isolated.

As a preliminary solution for these errors, we applied a filter on the duration of the samples to eliminate samples that are too short (usually, we do not consider syllables shorter than $10ms$) or too long (usually, we do not consider syllables longer than $300ms$). Of course, as for the selection parameters, these threshold values could change depending on the syllable type. This procedure allows to eliminate error due to a cutting failure and resulting in samples containing more than one syllable, samples containing a very short sound (which is not always a syllable), or, occasionally, the wrong, misclassified, syllable. Using a threshold based on the duration, we could remove 6672 errors. That is, we had an error equal to $8,46\%$ after the semi-automatic selection. The clean dataset contains 72155 syllables from the 16 classes. A random set of 100 single syllables (selected from the phrases) belonging to each class of the repertoire are collected in Figures 16 and 17.

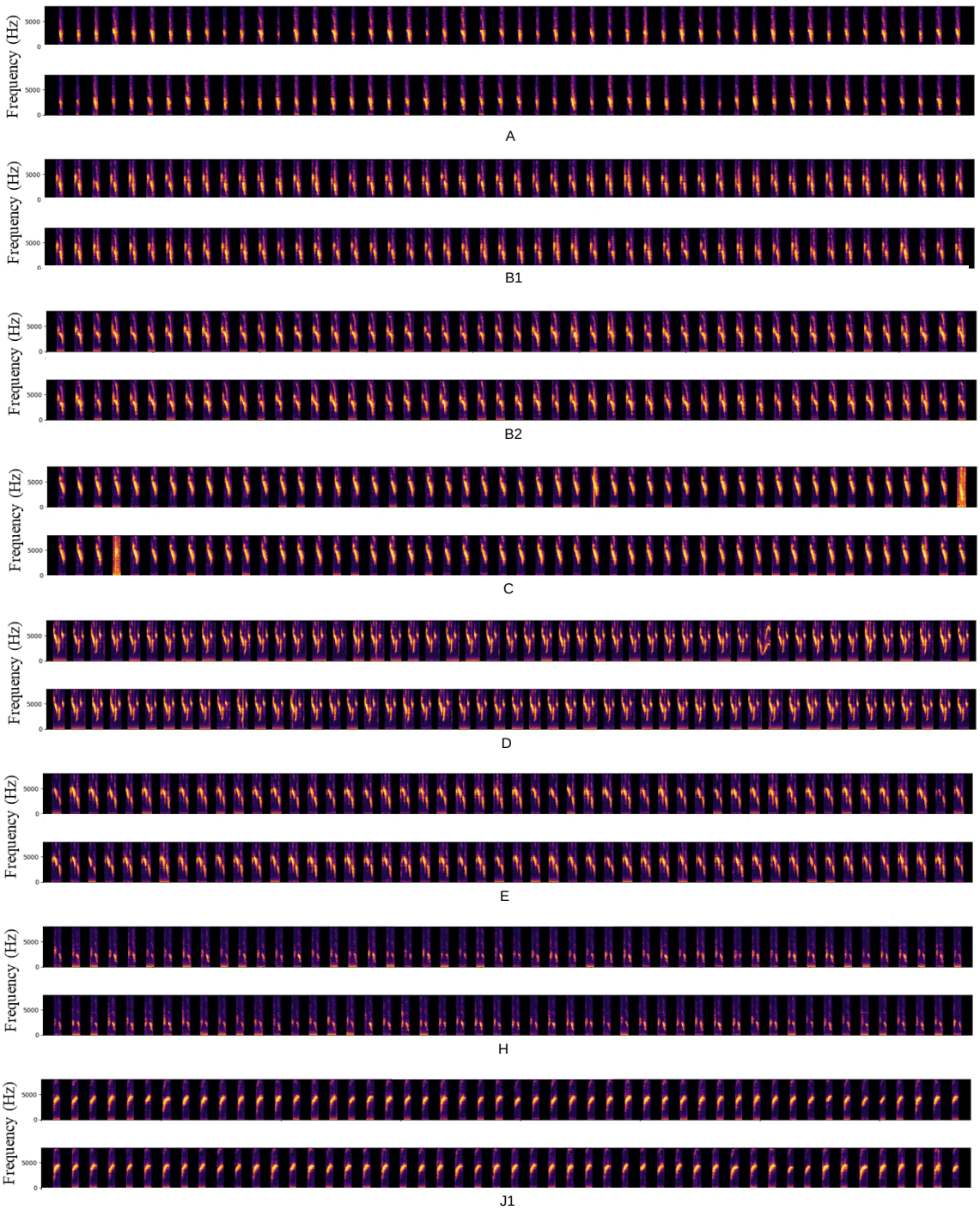


Fig. 16. **Example of single syllables.** For each class of the repertoire, 100 random selected syllables from the totality of the selected syllables. To select the syllables we used three parameters: amplitude threshold, minimal duration of the syllable and duration of the gap between two consecutive syllables. From the top: *A*, *B1*, *B2*, *C*, *D*, *E*, *H*, *J1*.

Errors due to an a priori misclassified phrase can't always be solved applying a threshold based on the syllable duration. On the one hand, if the difference between two syllables is evident, the error can be corrected simply using a threshold based on the syllable duration. This is the case of a phrase M wrongly classified as phrase A. The mean duration of syllable A is much smaller than the mean duration of syllable M: this means that we can fairly assume that a $100ms$ sample can't be syllable A (which has a shorter average duration). Vice versa, we can assume that a $30ms$ sample does not belong to class M. On the other hand, if two syllables share their duration distribution, it becomes more difficult to get rid of samples coming from a wrongly classified phrases. This is the case of a phrase M wrongly classified as phrase D: the two syllables have not

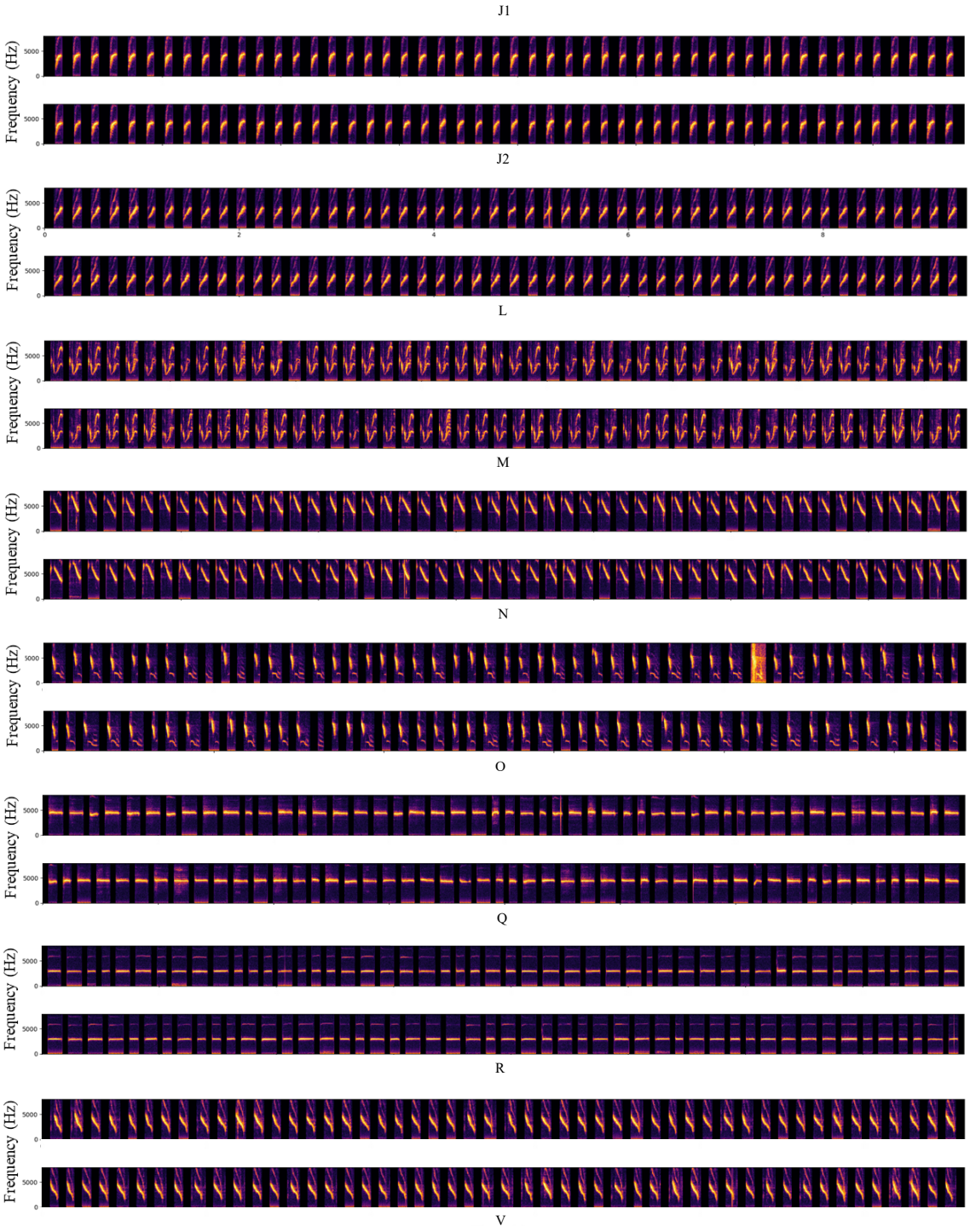


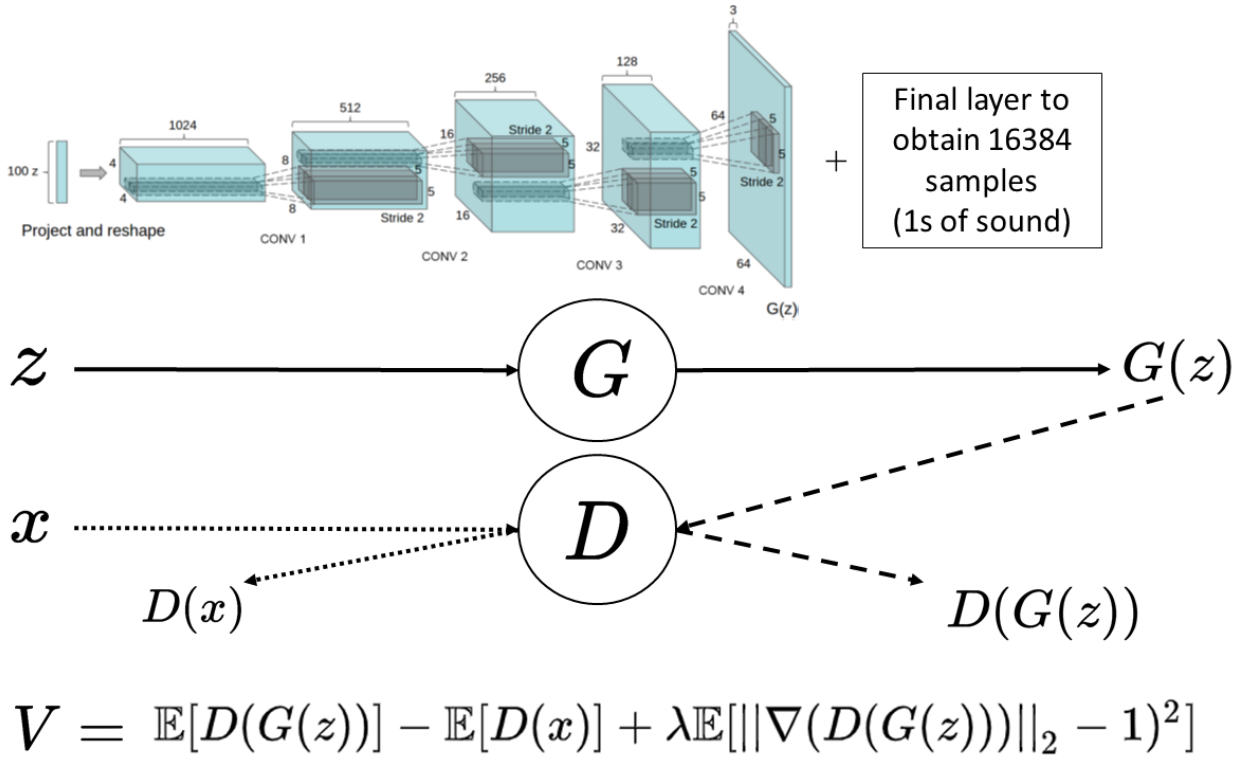
Fig. 17. **Example of single syllables: .** For each class of the repertoire,100 random selected syllables from the totality of the selected syllables. To select the syllables we used three parameters: amplitude threshold, minimal duration of the syllable and duration of the gap between two consecutive syllables. From the top: *J2*, *L*, *M*, *N*, *O*, *Q*, *R*, *V*.

only a similar duration distribution, but also a similar structure (Figure 16, panel *D*). To understand if this type of error can affect our work, we need to think about the use we want to make of the training dataset (e.g. which network we want to train and its characteristics). On the one side, we aim to use the dataset to train the generator of the GAN to produce realistic samples: from the structure of the network, we know that the generator does not care about the class of each samples (indeed, it does not know the distribution of the training data). Since the generator does not have access to the classification, it is not a problem to have well selected samples in the wrong class. On the other side, we want to use the dataset to train a classifier able to determine for each sample the class it belongs to. We need to have a clean dataset to be able to teach the classifier.

Nevertheless, after a visual inspection of the samples and after a validation of the classifier using the training dataset, we can assume that this type of error represent a low percentage of the total amount of error.

II. Appendix II: WaveGAN architecture

Figure 18 shows the structure of the generator G and the discriminator D, highlighting the architecture, the input and the output of the models, and the value function definition (i.e., V). The training data are pre-processed and stored in a tuple of *np.float32* tensors representing audio waveforms (x in Figure 18). The generator model G takes as input the latent vector $z \sim U(-1, 1)$. In the original paper, z is an 100-dimensional vector, but we will use several lower dimensional inputs in this study. The upper part of Figure 18 shows the architecture of G: it has been taken from DCGAN [11] generator and it has been modified with an additional layer to obtain as output 16384 samples (i.e., 1 s of sound). Similarly, the discriminator model D takes as input vectors of length 16384 and gives as output an object of shape $(16, 1024)$. D can take as input the training data x or the output of the generator $G(z)$, respectively resulting, after deconvolution, in $D(x)$ and $D(G(z))$. These two quantities are the variables of the value function V. To define V, Donahue et al. [15] removed the batch normalization from the generator and discriminator, and they used WGAN-GP [26] strategy during training. This strategy consists in the introduction of a gradient penalty term in the loss function, driven by the regularization hyperparamaeter λ.



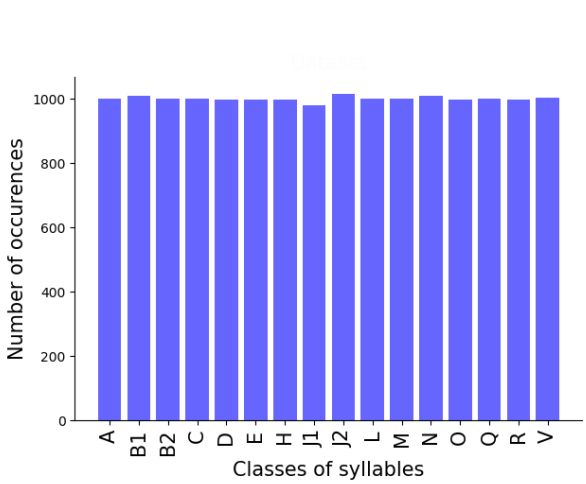
$$V = \mathbb{E}[D(G(z))] - \mathbb{E}[D(x)] + \lambda \mathbb{E}[||\nabla(D(G(z)))||_2 - 1)^2]$$

Fig. 18. **WaveGAN architecture**. The architecture of the generator model G is the same architecture used in DCGAN [11] with an additional convolutional layer to obtain 16384 samples (i.e., 1 s of sound). The generator takes a latent vector $z \sim U(-1, 1)$ as input. The discriminator model D takes alternatively the training data x and the output of the generator $G(z)$ as input. After deconvolution, a representation of shape $(16, 1024)$ is obtained. The outputs of the discriminator, $D(x)$ and $D(G(z))$ are used as variable of the value function V [26]. Image adapted from [11]

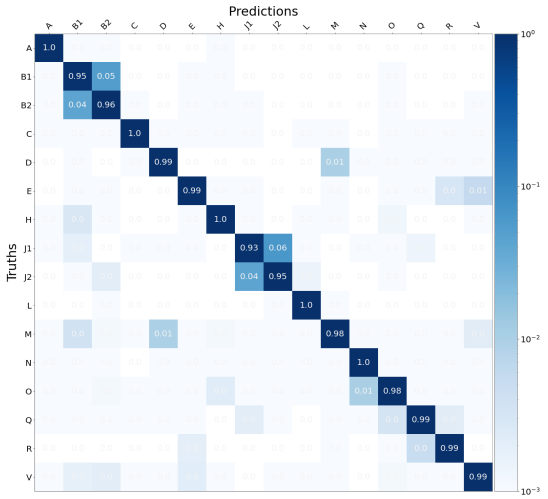
III. Classifier

A. Analysis of the training dataset using classifier-REAL

As for *classifier-EXT* (Figure 2a), the average number of syllables recognized for each of the 16 classes of the repertoire is close to $1k$ for. This is coherent with the fact that the training dataset contains $1k$ samples per class. Moreover, a confusion between syllable $B1$ and syllable $B2$, and between syllable $J1$ and syllable $J2$ can be seen on the diagonal in correspondence of these elements (sub-diagonal elements are darker than the others for these 4 syllables).



(a) Distribution of the training data obtained *classifier-REAL*.



(b) Confusion matrix *classifier-REAL*.

Fig. 19. **Training data analysis with *classifier-REAL***. Panel (a) shows the distribution obtained using *classifier-REAL* on the training dataset. The average number of syllables per class is $1k$ for each of the 16 classes of the repertoire. Panel (b) shows the confusion matrix (i.e. the level of confidence of the classifier in making the correct assignment) relative to *classifier-REAL*. Each square represent the level of confidence of the classifier in making the correct assignment. The level of confidence is expressed on a logarithmic scale using shades of blue.

B. Robustness of the classifier

The evaluation of the robustness of *classifier-REAL* and *classifier-EXT* has been performed using a 10 folds cross-validation over the three corresponding training datasets. For each fold, 5 different instances of each classifier were randomly initialized, trained and tested. The models were evaluated using an accuracy measure.

The accuracy has been defined as:

$$\text{accuracy}(y, \hat{y}) = \frac{1}{n_{timesteps}} \sum_{i=0}^{n_{timesteps}} \mathbb{1}(y(i) - \hat{y}(i)) \tag{4}$$

where $\mathbb{1}(x)$ is the indicator function, and considering a sequence of data x of length $n_{timesteps}$, a sequence of labels y and a sequence of classifier outputs $\hat{y}$, both also of length $n_{timesteps}$. The accuracy represents the capability of the classifier of making correct assignments for each time steps of MFCCs encoding the audio signal. An accuracy score close to 1 therefore indicates that the classifier is able to recognize the syllable category of each sample and to correctly assign this category to each time steps representing this sample.

Both *classifier-REAL* and *classifier-EXT* show an high level of accuracy (Figure 20). As summarized in Table II the mean accuracy for the validation set is 0.9815 ± 0.0024 for *classifier-REAL* and 0.9756 ± 0.0025 for *classifier-EXT*.

	Mean accuracy	
Classifier	Train	Validation
classifier-REAL	0.9832 ± 0.0005	0.9815 ± 0.0024
classifier-EXT	0.9777 ± 0.0006	0.9756 ± 0.0025

TABLE II

Mean accuracy. Comparison between *classifier-REAL* and *classifier-EXT* in terms of the median of the accuracy. For each model, the accuracy has been computed for the training set and for the validation set. The highest mean value has been reached with *classifier-REAL* where no alternative unknown classes have been introduced.

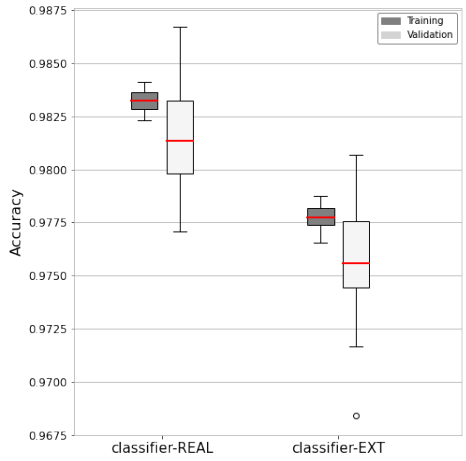
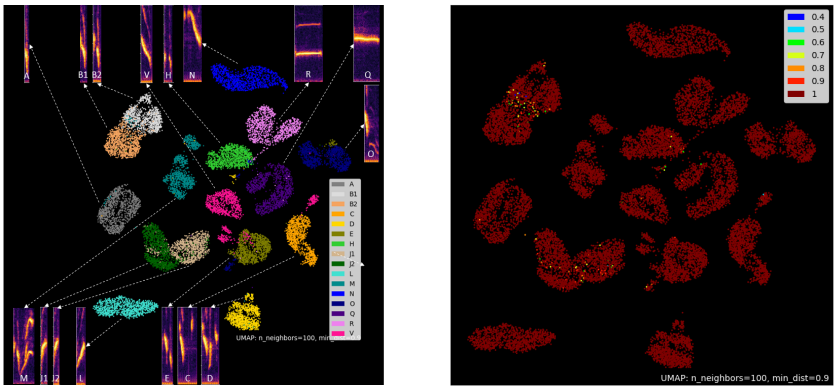


Fig. 20. **Accuracy of the classifiers.** Comparison between *classifier-REAL* (in the left) and *classifier-EXT* (on the right) in terms of the accuracy. For each model, the accuracy has been computed for the training set (gray rectangles) and for the validation set (white rectangles). The red lines represent the median accuracy relative to each set for each model. The white point visible for *classifier-EXT* represents an outlier, determined by the fact that the accuracy distribution is sharp. The highest accuracy has been reached with *classifier-REAL* where no alternative unknown classes have been introduced.

C. Certainty of the classifier

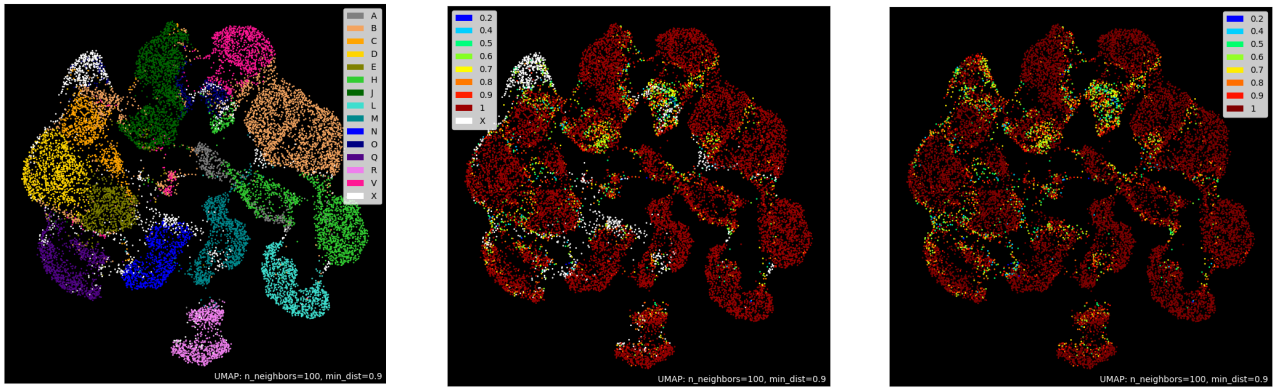
For each syllable, the classifier described in Section III-D provides a distribution which determines to which class the classifier assign that particular syllable. As mentioned, a soft-max operation is then applied in order to obtain a distribution bounded between 0 and 1. That is, each syllable is assigned to a particular class with a probability given by the max value of the resulting N-dimensional vector, where N is the number of classes present in the vocabulary. For instance, $N = 21$ for *classifier-EXT*. Such a classifier gives high scores for the training data, for which the majority of the syllables is assigned to a class with $p_s > 0.9$ (brown points in Figure 21b). Only a few syllables are assigned to a class with a probability $p_s \leq 0.9$.

Differently, *classifier-EXT* shows a higher uncertainty while classifying the generated data (Figure 22b). Although the majority of the syllables is assigned to a class with $p_s > 0.9$ (brown points), the number of syllables assigned to a particular class with a lower probability increases. Moreover, a higher uncertainty is often connected with the presence of syllables assigned to class *X* (left panel of Figure 22b).



(a) Vocabulary (b) Intensity

Fig. 21. **Certainty of the classifier: training dataset**. Panel (a) shows representation of the training data as in Figure 1. Each point corresponds to a spectrogram, and each color represents a class of the repertoire. Panel (b) shows the probability of each syllable (each point) to belong to a particular class of the repertoire. Each point corresponds to a spectrogram, and each color corresponds to an interval of probability starting at the value indicated in the legend and ending at the next color value. For example, brown points belong to a certain class of the vocabulary with a probability $0.9 < p_s \leq 1$. An higher uncertainty of *classifier-EXT* can be observed when it is applied to the generated data: indeed, an higher number of syllables are assigned to a certain class with a probability $p_s \leq 0.9$ (all the points but the brown points).



(a) Vocabulary (b) Intensity

Fig. 22. **Certainty of the classifier: generated data**. Panel (a) shows representation of the generated data as in Figure 9. Each point corresponds to a spectrogram, and each color represents a class of the vocabulary (repertoire and class *X*). Panel (b) probability of each syllable (each point) to belong to a particular class of the repertoire (right panel), with class *X* highlighted in white (left panel). Each point corresponds to a spectrogram, and each color corresponds to an interval starting at the value indicated in the legend and ending at the next color value. For example, brown points belong to a certain class of the vocabulary with a probability $0.9 < p_s \leq 1$. An higher uncertainty of *classifier-EXT* can be observed when it is applied to the generated data: indeed, an higher number of syllables are assigned to a certain class with a probability $p_s \leq 0.9$ (all the points but the brown points).

IV. Extension of the qualitative analysis

A. Generations across time

Complete version of Figure 3 generations across time for each class of the repertoire at epochs 0,15, 45, 106, 514 and 984.

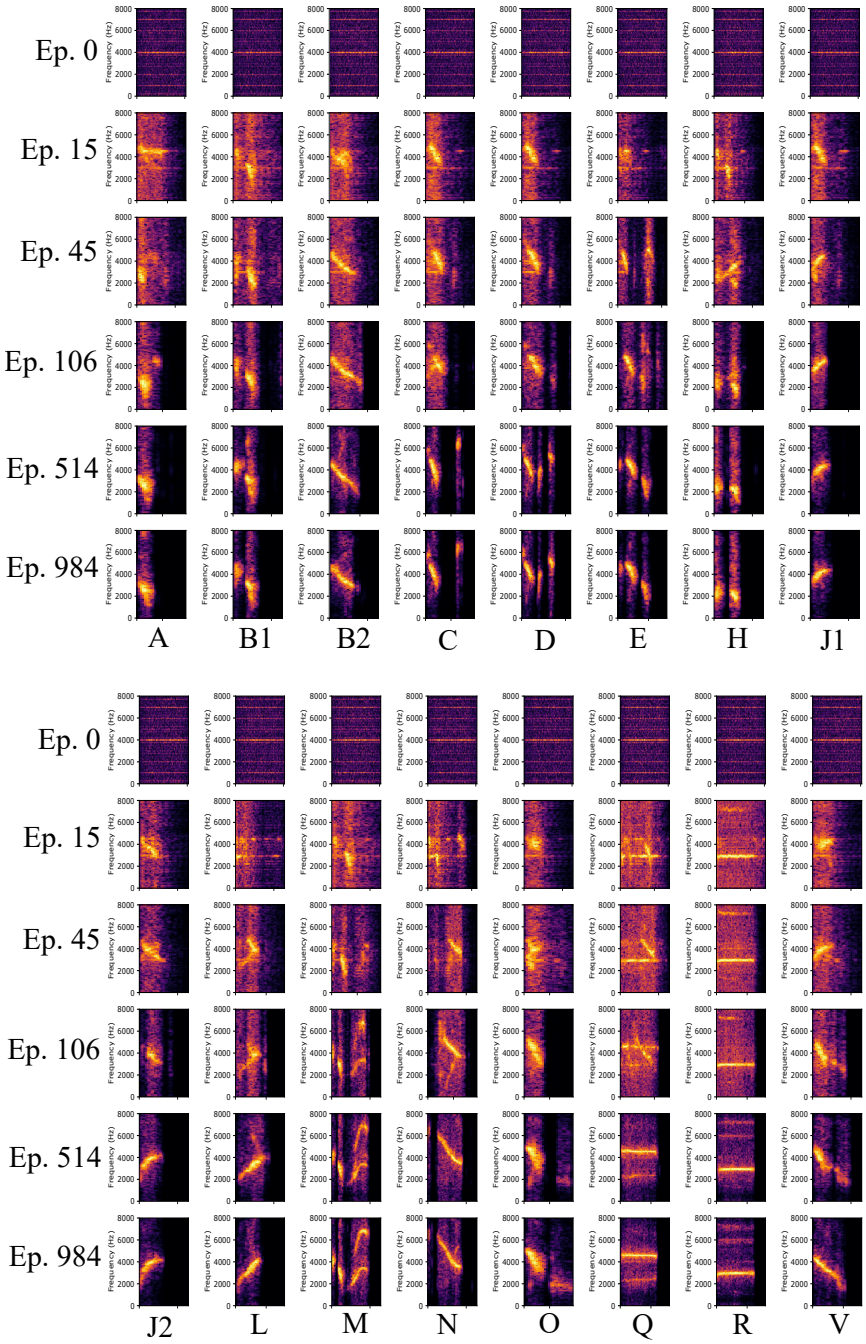


Fig. 23. **Generations across time.** Example of one selected syllable per class across time. Each syllable has been first generated at epoch 984 and recognized using *classifier-EXT*. Then, the latent vector associated at each syllable has been used to generate the same syllable at epochs 0,15, 45, 106 and 514. At epoch 0 the generations do not vary from one class to another. At epoch 15 the generations start to be coherent in duration but remain noisy and unclear. At epoch 45 some syllables are more distinguishable than at earlier epochs, but the majority remains noisy and indistinguishable. As the training goes on, the generations resemble more and more the real syllables (epochs 106 and 514). The generations obtained at epoch 984 have a clear distinguishable shape. Here, the generator obtained from the instance *Ex 6* in Figure 5 has been used to generate the syllables across time; latent space dimension $ld = 3$; *Ep.* stands for epoch.

B. Mean spectrogtam across time

At early epochs of training, when not all the classes are covered by the generator (empty boxes in Figure 7(a-b)), the mean spectrograms are blurry and show syllables difficult to recognize as belonging to a class of the repertoire (see Figure **??**). The

spectrograms look often as a mix of syllables coming from several classes (e.g, syllable *A* or syllable *R* in Figures 7(a-b)). At epoch 514 (Figure 7c) all the syllables can be produced by the generator and only a few syllables remain difficult to be recognized as belonging to a class of the repertoire (e.g., syllable *B2* and syllable *L*). Nevertheless, a lot of syllables are clearly recognizable (e.g., syllable *N* and syllable *Q*). Finally, at advanced stages of training, all the spectrograms show a recognizable syllable (epoch 984).

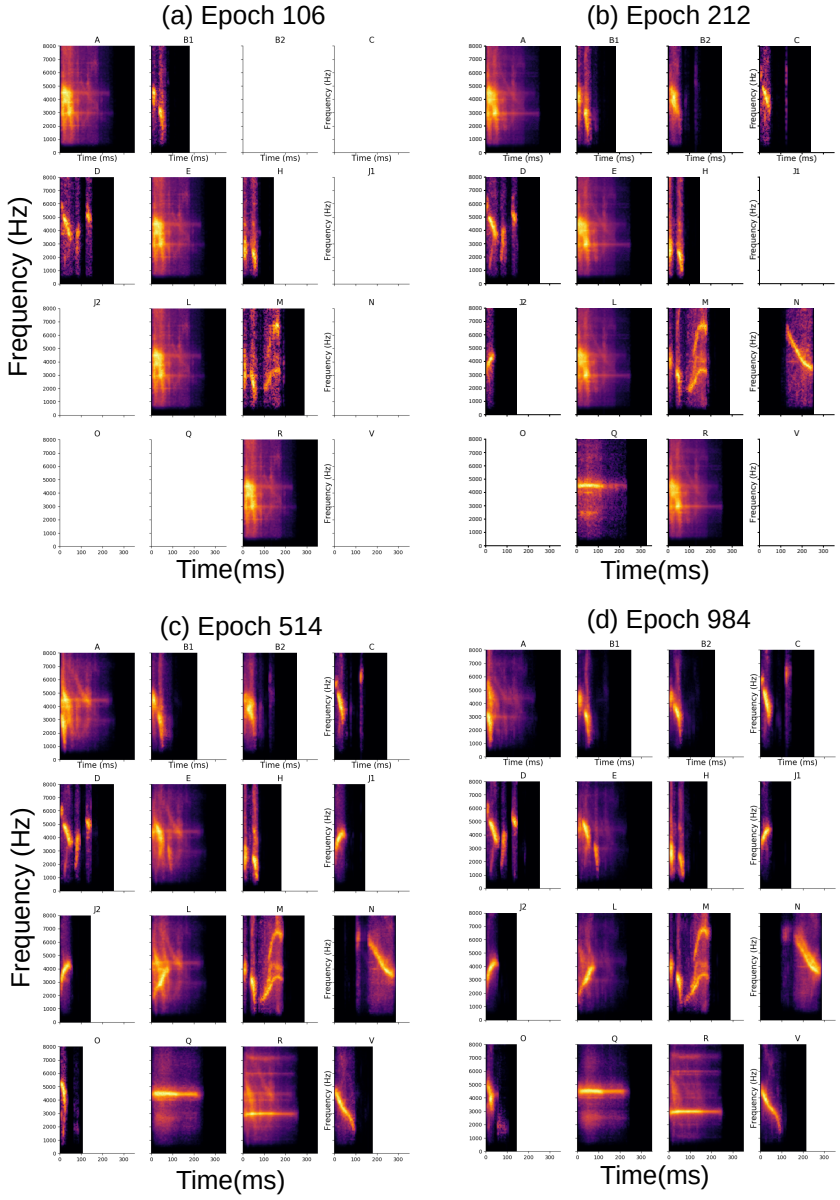


Fig. 24. **Mean spectrogram across time.** Mean spectrogram of $1k$ the syllables generated at epoch 106 (a), 212 (b) 514 (c) and 984 (d). Empty boxes in panels (a) and (b) mean that at epoch 106 and 212 not all the repertoire can be covered by the generator and no syllables have been recognized by *classifier-EXT* as elements of the not represented classes (e.g., class *B2*). At epochs 106 (panel (a)) and 212 (panel (a)) the correct duration and, eventually, the content of the syllables can be grasped. At epoch 514 (panel (c)) almost all the syllables can be distinguished and only a few remain noisy and unclear (*A*, *E*, *L*, *R*). At epoch 984 (panel (d)) all the syllables are clear and distinguishable. They can be compared with the repertoire in Figure **??**. Here, the training has been done using $ld = 3$, the generator obtained from the instance *Ex 6* in Figure 5 has been used to generate the syllables across time and *classifier-EXT* has been used to identify the generated syllables.

C. UMAP representation

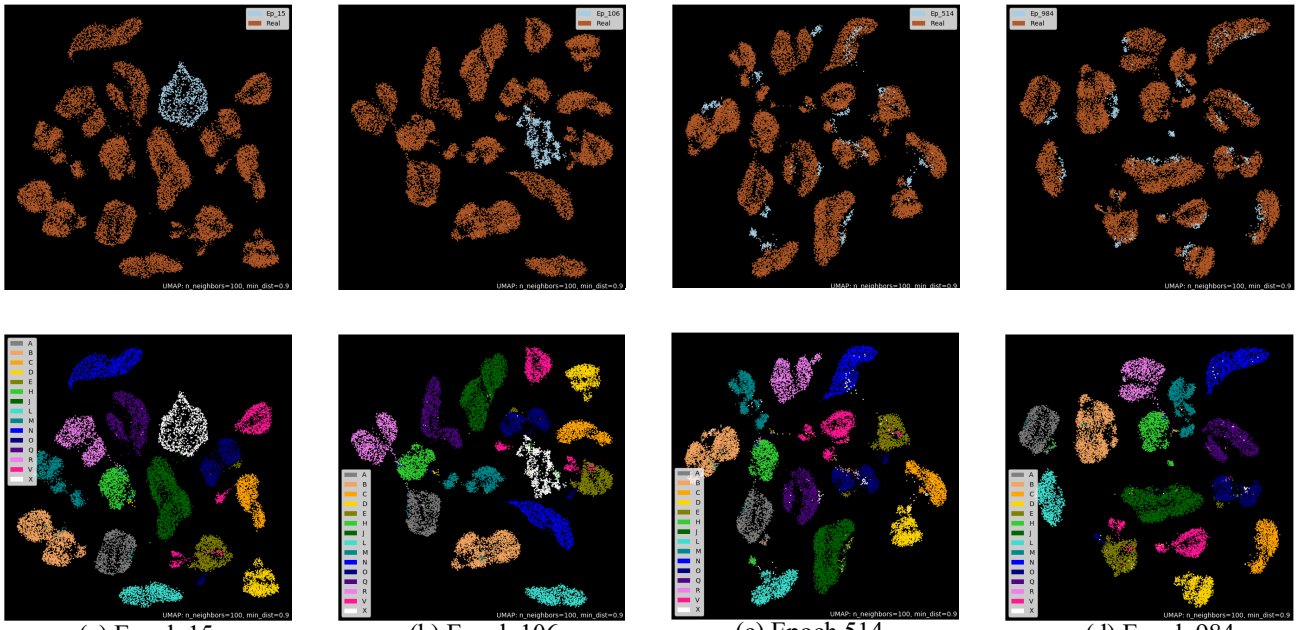
Complete version of Figure 8. The UMAP representation of the generated data and the training data show that (1) the generated data are grouped together as they were an additional cluster with respect to the ones obtained from the training data (upper panels of Figure 25) and (2) the generated data belong to the same cluster of the training data (bottom panels of Figure 25). On the one hand, the generated data spread over time, moving from being a cluster in itself (light blue points in the upper panels of Figure 25) to taking a conformation compatible with the training data (brown points in the upper panels

TABLE III

HUMAN JUDGMENT. COHEN'S KAPPA COEFFICIENT COMPUTED PER EACH COUPLE ACROSS HUMAN JUDGES AND VERSUS *classifier-EXT*. EACH JUDGE EVALUATED 200 SYLLABLES PRODUCED USING THE GENERATOR OF INSTANCE *Ex 6*. THE KAPPA COEFFICIENTS κ_{Cohen} OBTAINED ACROSS JUDGES AND WITH RESPECT TO *classifier-EXT* ARE COMPARABLE.

	Cohen's kappa
Judge 1 vs Judge 2	$0,74$
Judge 2 vs *classifier-EXT*	$0,79$
Judge 1 vs *classifier-EXT*	$0,73$

of Figure 25). On the other hand, the generated data are mostly constituted by syllables belonging to class *X* (the cumulative class of the alternative unknown classes (in this case, *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*) at early stages of training (bottom panels of Figure 25). Later, the majority of the generated data belongs to the same class as the closer cluster of training data in the UMAP representation.



(a) Epoch 15. (b) Epoch 106. (c) Epoch 514. (d) Epoch 984.

Fig. 25. **Syllable space representation across time.** Syllable space representation obtained from the training dataset ($16k$ syllables) and $1k$ syllables generated at epochs 15 (a), 106 (b), 514 (c) and 984 (d) using UMAP [31]. Upper panels show the training data (brown points) and the generated data (blue points). These four figures are different because the analyzed dataset differs for the $1k$ generations specific of each epoch. Bottom panels show the same representation with the classes of syllables highlighted with different colors. Each cluster/color corresponds to one class of the repertoire and class *X* (in white) represents the cumulative class of the alternative unknown classes (in this case, *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*). Here, the training has been done using $ld = 3$, the generator of instance *Ex 6* has been used to generate the syllables and *classifier-EXT* has been used to identify both the generated data and the training data.

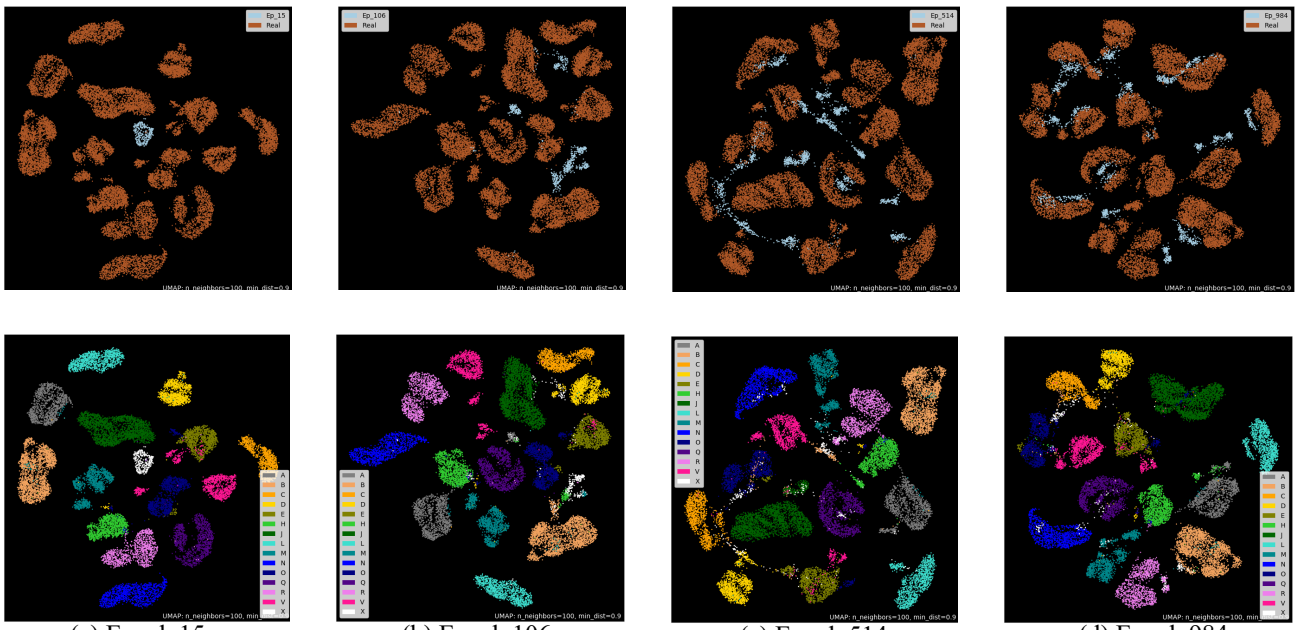
A similar UMAP representation to the one shown in Figure 25 can be obtained considering a balanced subset of ~ 2100 generated syllables (100 per class) instead of $1k$ random generations at epochs 15, 106, 514 and 984.

The fact that the generated syllables seem to be located at a bigger distance from the training data (Figure **??**(a-c, g)) when considering a balanced dataset of 2100 samples (with respect to the same representation obtained from $1k$ random generated syllables shown in Figure 8(a-c, g)) can be related to the way UMAP determines the clusters. Indeed, as much generated data are given to UMAP as much the representation obtained takes into account the difference between the real and the generated data, showing a less compact representation.

D. Human judgjement

E. Stability of the training

The UMAP representations (Figure 27) and the mean spectrograms (Figure 28) show a similar representation for consecutive epochs (i.e., epochs 969, 984 and 999) which show the stability of instance *Ex 6* around the convergence of the training. The differences in the UMAP representation (Figure 27) are given by the fact that the set of generated data is different at each epoch.



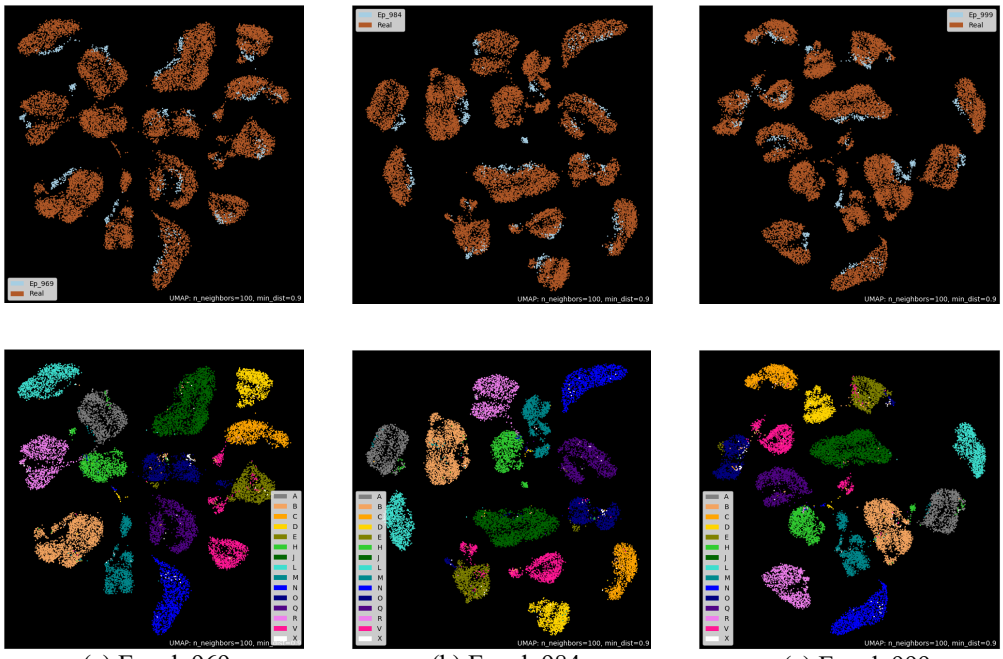
(a) Epoch 15. (b) Epoch 106. (c) Epoch 514. (d) Epoch 984.

Fig. 26. **Syllable space representation across time: balanced representation.** Syllable space representation obtained from the training dataset ($16k$ syllables) and 2100 syllables (100 per class, when the class is present, for a total of 21 classes - the 16 classes of the repertoire and 5 alternative unknown classes, here grouped as class *X*) generated at epochs 15 (a), 106 (b), 514 (c) and 984 (d) using UMAP [31]. Upper panels show the training data (brown points) and the generated data (blue points). These four figures are different because the analyzed dataset differs for the 2100 generations specific of each epoch. Bottom panels show the same representation of panels with the classes of syllables highlighted with different colors. Each cluster/color corresponds to one class of the repertoire and class *X* (in white) represents the cumulative class of the alternative unknown classes (in this case, *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*). Here, the training has been done using $ld = 3$, the generator of instance *Ex 6* has been used to generate the syllables and *classifier-EXT* has been used to identify both the generated data and the training data.

F. Latent space exploration

Figures 30 and 31 shows a similar analysis as the one in Figure 11. Here, the second and the third component of the latent spac have been varied using a variational step equal to $v_{step} = 0.001$ and a syllable has been generated at each step. The UMAP representation of the newly generated syllables and the generated dataset shown in Figure 9 shows a continuity in the productions of the generator (light blue lines in the left panels of Figure 32(a-b)).

Moreover, the continuity of the latent space can be observed when observing the transition from a syllable to another, as shown in Figure 12b. Figure 33 shows two more example of transition are shown: (i) from syllable *M* tp syllable *V* and (ii) from syllable *H* to syllable *N*. The transition from syllable *H* to syllable *N* shows (1) an interesting stretch of syllable *B* (cream cluster in the right panels of Figure 33) connecting syllable *B* and syllable *Q* and (2) the uncertainty of the classifier in differentiating between syllable *H* (light green cluster in the right panels of Figure 33) and syllable *A* (gray cluster in the right panels of Figure 33).



(a) Epoch 969. (b) Epoch 984. (c) Epoch 999.

Fig. 27. **Stability of UMAP representation.** UMAP representation of the training dataset ($16k$ syllables) and $1k$ generated data at epochs 969 (a), 984 (b) and 999 (d). The upper panels show the representation of the training data (brown points) and the generated data (light blue points) over time. The bottom panels show the same representation highlighting the classes of the vocabulary. Each cluster/color correspond to one class of the repertoire and class X (in white) represents the cumulative class of alternative unknown classes (in this case, *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*). Although the representation is slightly different (given the fact that the generated syllables vary across time,it is possible to observe how the generated syllable mix well with the clusters obtained from the training data at all epochs), it remains stable across consecutive epochs. Here, the training has been done using $ld = 3$, the generator of instance *Ex 6* has been used to generate the syllables and *classifier-EXT* has been used to identify both the generated data and the training data.

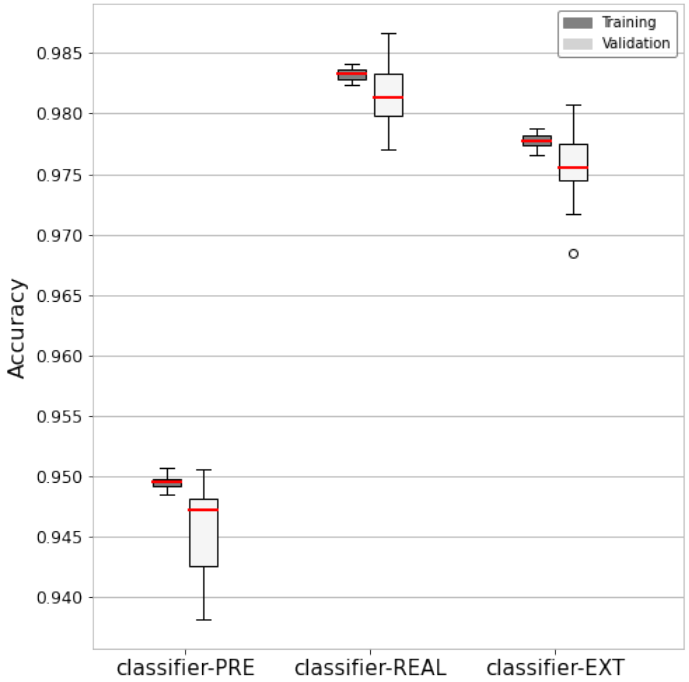


Fig. 28. **Stability of the mean spectrogram.** Mean spectrogram of $1k$ syllables generated at epochs 969 (a), 984 (b) and 999 (c). The three epochs share a similar representation of the syllables. Here, the training has been done using $ld = 3$, the generator of instance *Ex 6* has been used to generate the syllables and *classifier-EXT* has been used to identify the generated data.

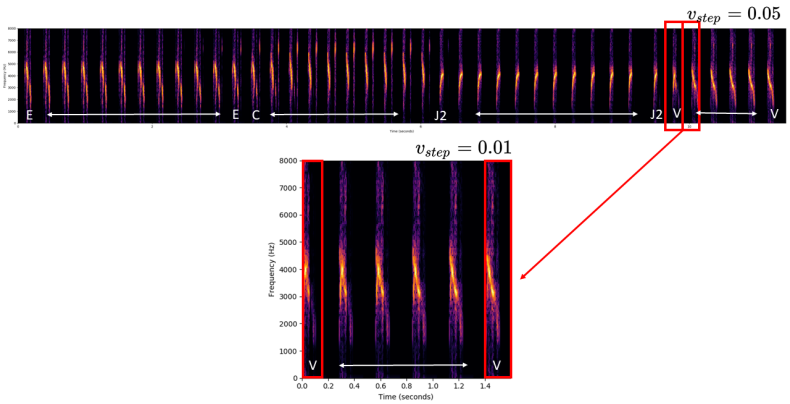


Fig. 29. **Exploration of the latent space: one component variation. First component.**. We selected a random latent vector $z \sim R^3([-1, 1])$ to create a baseline syllable. Then, we moved one by one the components of the vector by a variation step equal to $v_{step} = 0.05$. We observed all the syllables produced to look at how they evolve and if there are non-smooth transitions. The upper panel of Figure 11 shows the exploration of the first component of the latent vector obtained with $v_{step} = 0.05$. The syllable *V* contained in the red square on the right represent a point where a non-smooth transition has been detected. For these particular transitions, we considered the two consecutive syllables obtained from the first variation step (i.e., two consecutive syllables contained in two consecutive red squares) and we applied a variation step of $v_{step} = 0.01$ to the first component of the latent vector to generate intermediate syllables. The bottom panel show the exploration of the non-smooth transition highlighted above. The syllables have been obtained the 3-dimensional generator obtained from instance *Ex 6* and the name of the syllable for this analysis has been provided by *classifier-EXT*.

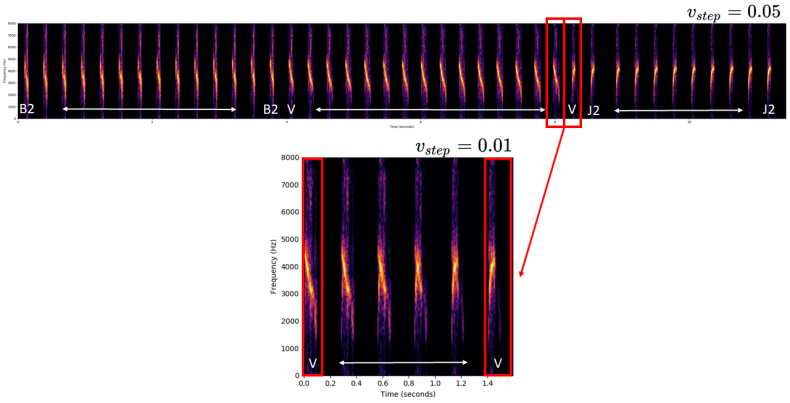


Fig. 30. **Exploration of the latent space: one component variation. Second component.**. We selected a random latent vector $z \sim R^3([-1, 1])$ to create a baseline syllable. Then, we moved one by one the components of the vector by a variation step equal to $v_{step} = 0.05$. We observed all the syllables produced to look at how they evolve and if there are non-smooth transitions. The upper panel shows the exploration of the second component of the latent vector obtained with $v_{step} = 0.05$. The syllable *V* contained in the red square on the right represent a point where a non-smooth transition has been detected. For these particular transitions, we considered the two consecutive syllables obtained from the first variation step (i.e., two consecutive syllables contained in two consecutive red squares) and we applied a variation step of $v_{step} = 0.01$ to the first component of the latent vector to generate intermediate syllables. The bottom panel show the exploration of the non-smooth transition highlighted above. The syllables have been obtained the 3-dimensional generator obtained from instance *Ex 6* and the name of the syllable for this analysis has been provided by *classifier-EXT*.

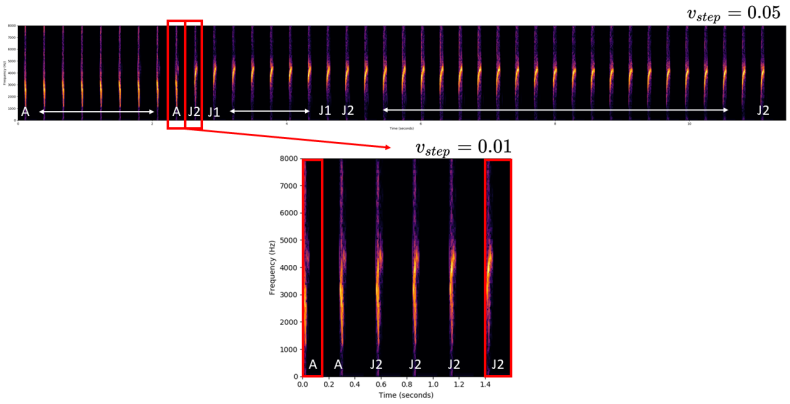
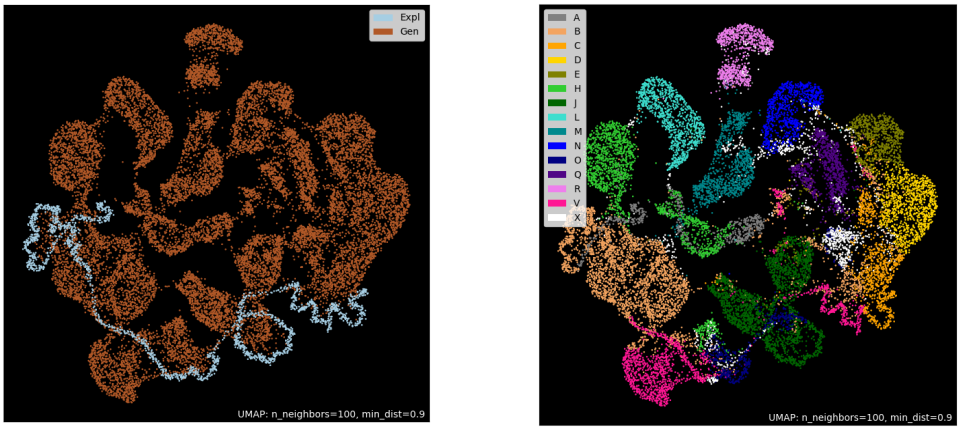


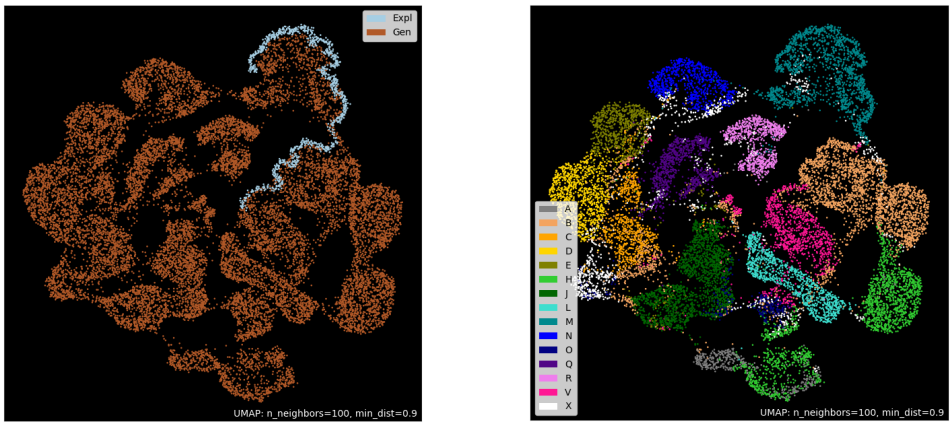
Fig. 31. **Exploration of the latent space: one component variation. Third component.**. We selected a random latent vector $z \sim R^3([-1, 1])$ to create a baseline syllable. Then, we moved one by one the components of the vector by a variation step equal to $v_{step} = 0.05$. We observed all the syllables produced to look at how they evolve and if there are non-smooth transitions. The upper panel shows the exploration of the third component of the latent vector obtained with $v_{step} = 0.05$. The syllable *V* contained in the red square on the right represent a point where a non-smooth transition has been detected. For these particular transitions, we considered the two consecutive syllables obtained from the first variation step (i.e., two consecutive syllables contained in two consecutive red squares) and we applied a variation step of $v_{step} = 0.01$ to the first component of the latent vector to generate intermediate syllables. The bottom panel show the exploration of the non-smooth transition highlighted above. The syllables have been obtained the 3-dimensional generator obtained from instance *Ex 6* and the name of the syllable for this analysis has been provided by *classifier-EXT*.



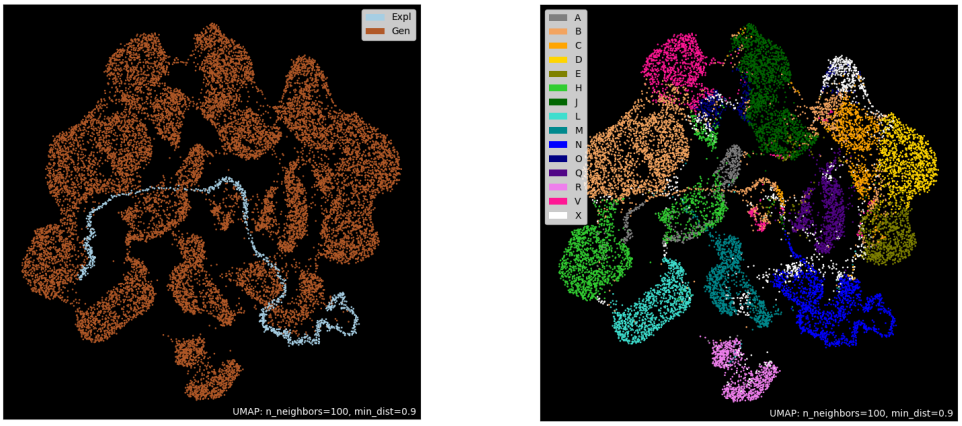
(a) Variation in the second component

(c) Variation in the third component

Fig. 32. **Latent vector components variation.** Panel (a) represents the first component variation, panel (b) represents the second component variation and panel (c) represents the third component variation. The left panels show the variational data (generated at each step) (light blue points) and $16k$ generated data from the same model at the same epoch (brown points). Each cluster/color in the right panels corresponds to one class of the repertoire and class *X* (in white) represents the cumulative class of the alternative unknown classes (in this case, *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*). Here, the training has been done using $ld = 3$. A latent vector has been randomly selected and a step $v_{step} = 0.001$ has been applied to its component, one by one. The generator obtained from instance *Ex 6* of the training has been used *classifier-EXT* has been used to identify the generated syllables from the generator of instance *Ex 6*.



(a) Transition from M to V



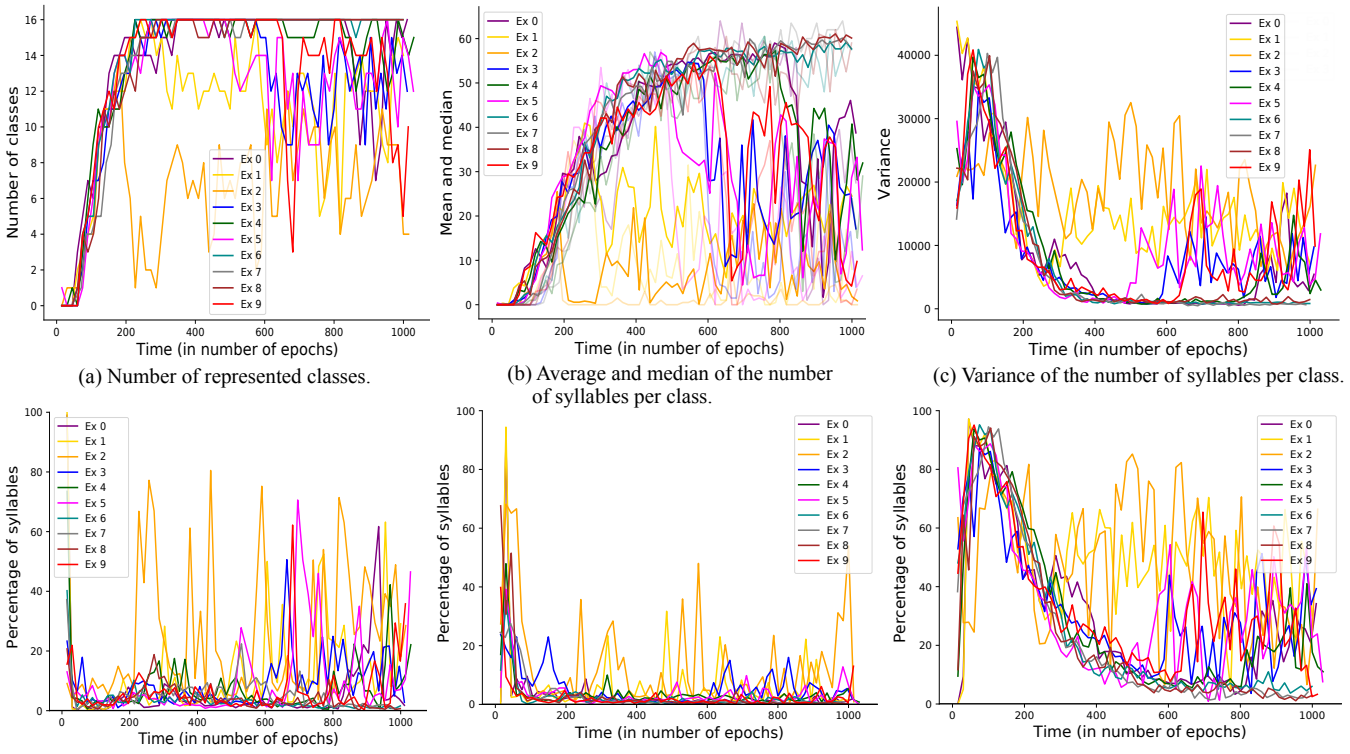
(b) Transition from H to N

Fig. 33. **Transition between two generated syllables.** Panel (a) represents the transition between *M* (turquoise cluster) and syllable *D* (yellow cluster), panel (b) the transition between *M* (turquoise cluster) and syllable *V* (magenta cluster) and panel (c) the transition between *H* (light green cluster) and syllable *N* (blue cluster). The left panels show the variational data (generated at each step) (light blue points) and $16k$ generated data from the same model at the same epoch (brown points). Each cluster/color in the right panels corresponds to one class of the repertoire and class *X* (in white) represents the cumulative class of the alternative unknown classes (in this case, *EARLY15*, *EARLY30*, *EARLY45*, *OT* and *WN*). Here, the training has been done using $ld = 3$ and *classifier-EXT* has been used to identify the generated syllables from the generator of instance *Ex 6*.

V. Extension of the quantitative analysis

A. *Different instances of training of a 3-dimensional WaveGAN*

The analysis of 10 instances of training of a 3-dimensional WaveGAN shows that not all the instances have the capability of providing a generator able of producing syllables belonging to all the classes of the repertoire (Figure 34a). Instances *Ex 2* (orange line), *Ex 1*(yellow line) and *Ex 5* (magenta line) show an early drop and, eventually, never reach to cover the repertoire. Nevertheless, the other instances show a drop at an advanced stage of the training (i.e., after epoch 600) or never drop. Similarly, the instances showing instability in Figure 34a show instability in the average number of syllables recognized per class (Figure 34b) and in the variance of the number of syllables recognized per class (Figure 34c). Moreover, alternative unknown classes, and in particular class *EARLY45*, are more represented even in advanced epochs of the training for those instances showing instability, as shown in Figures 34(d-f). Such an instability might characterize the beginning of overtraining: the generator starts to produce samples that are recognized as elements of an *EARLY* class or of class *OT*. The latter is more represented in instances showing overtraining (Figure 35). After, the generator is not able to recover. Further analysis are needed to understand how to carefully describe overtraining.



(a) Number of represented classes.

(b) Average and median of the number of syllables per class.

(c) Variance of the number of syllables per class.

(d) Number of syllables belonging to the *EARLY15* class.

(e) Number of syllables belonging to the *EARLY30* class.

(f) Number of syllables belonging to the *EARLY45* class.

Fig. 34. **Different instances of a 3-dimensional WaveGAN**. Statistical analysis performed on the classifier distribution of 10 instances of training. Panel (a) shows the number of classes represented by the generated data: that is, at each epoch, how many syllables of the repertoire are covered by the generator. An early drop (i.e., before epoch 600) in the number of represented classes can be seen for three instances (i.e, *Ex 2* – orange line, *Ex 1* – yellow line and *Ex 5* – magenta line). Panel (b) shows the average number (dark colored lines) of elements per class and the median (light colored lines): depending on the instance, the mean and the median could increase over time and remain stable (*Ex 6* – light blue line, *Ex 7* – gray line, *Ex. 8* – burgundy line), or increase until a certain epoch and then drop (*Ex 3*-blue line, *Ex 4* – green line, *Ex 5* – magenta lines, *Ex 9*-red line), or remain low for all the duration of the training (*Ex 1* – yellow line, *Ex 2* – orange line). To compute the quantities in panels (a) and (b), alternative unknown classes $x \in X$ have not been taken into account. Panel (c) shows the evolution of the variance of how many syllable per class have been produced. Here, $x \in X$ classes are included. For successful instances of training (*Ex 6* – light blue line, *Ex 7* – gray line, *Ex. 8* – burgundy line), the variance starts at a high value when the majority of the samples produced are not classified as syllables of the repertoire, then it decreases when the generator becomes better at producing syllables. Eventually, it increases again later (*Ex 3* – blue line, *Ex 4* – green line, *Ex 5* – magenta line, *Ex 9* – red line) or never decrease enough (*Ex 1* – yellow line, *Ex 2* – orange line). Panels (d-f) show the percentage of syllables that are classified as belonging to one of the alternative classes $x \in X$: from the left to the right, classes *EARLY15*, *EARLY30* and *EARLY45*. Alternative unknown classes, and in particular class *EARLY45*, are more present even in advanced epochs of the training for those instances where we identify overtraining (all but *Ex. 6* – light blue line, *Ex 7* – gray line and *Ex. 8* – burgundy line.

B. *Analysis of the parameters: complete version of the figures*

We used the training dataset described in Section III-A to train WaveGAN. Then, we used the experimental setup described in Section III-B. We used the *classifier-EXT* and *classifier-REAL* to identify the syllables as elements of the repertoire and 5 alternative unknown classes $x \in X$ (*EARLY15*, *EARLY30*, *EARLY45*, *OT*, and *WN*).

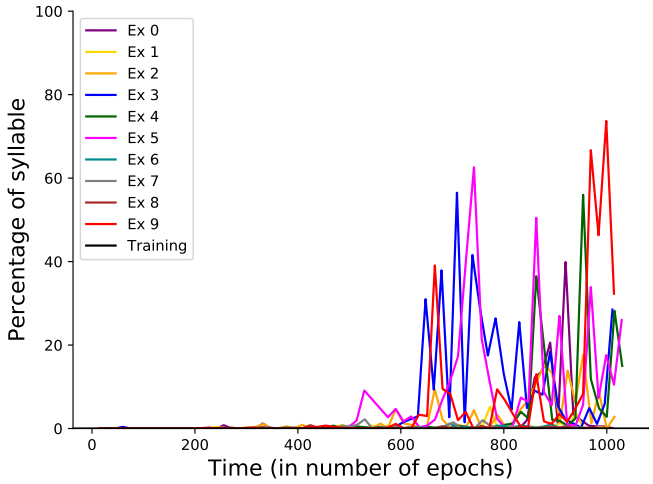
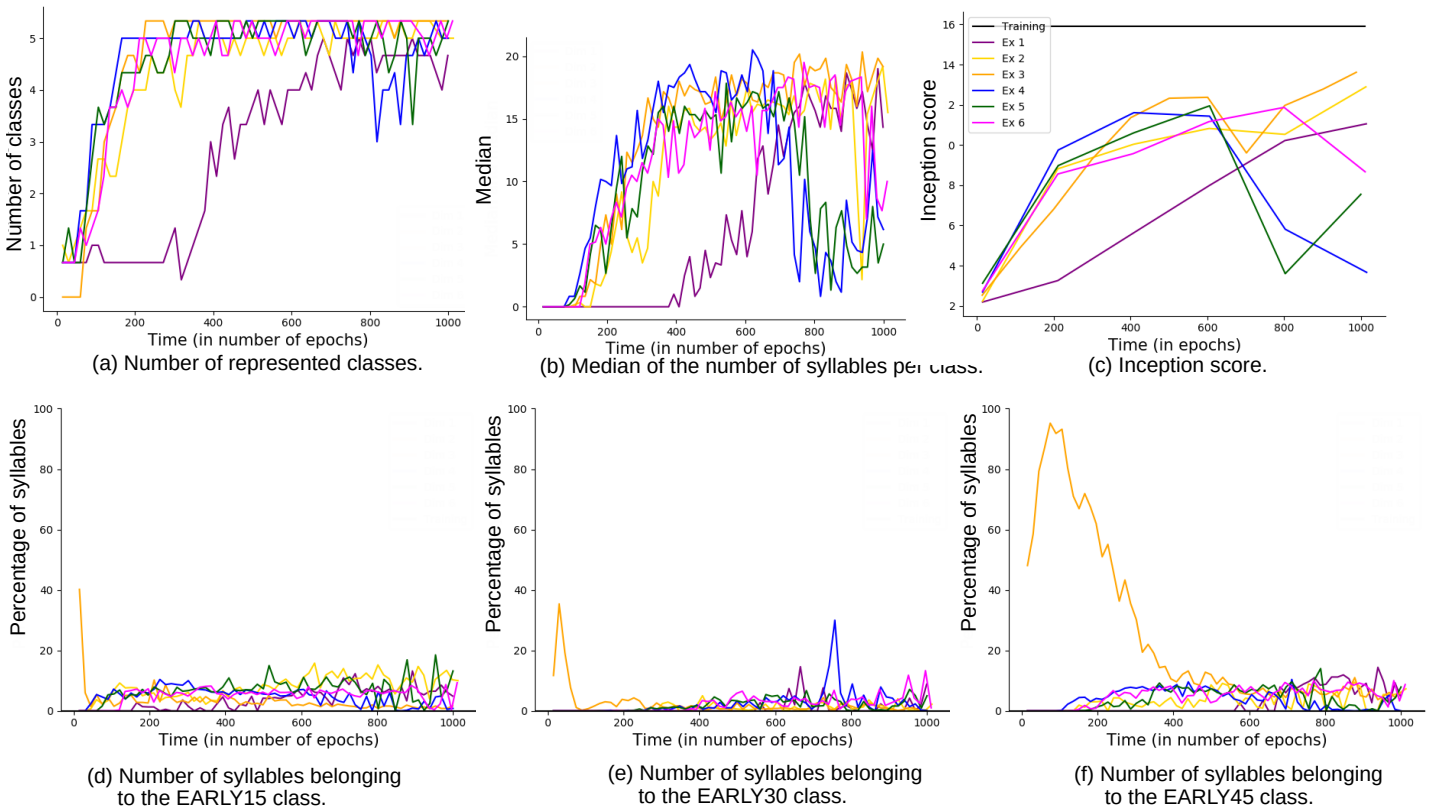


Fig. 35. **Overtraining**. Comparison between 10 instances of training: percentage of syllables classified by *classifier-EXT* as elements of class *OT* (i.e., overtraining). Instances *Ex 0* (purple line), *Ex 1* (yellow line), *Ex 2* (orange line), *Ex 3* (blue line), *Ex 4*(green line), *Ex 5* (magenta line), *Ex 9* (red line) show an increasing number of syllables classified as belonging to class *OT*. These instances show instability also in the average an variance of the syllables belonging to the classes of the repertoire and in the number of syllables belonging to an *EARLY* class (see Figure 34). Latent space dimension: $ld = 3$.

1) Latent space dimension: Complete version of Figures 13 (Figure 36). Accordingly to what observed in the main paper for Figure 36(a-c) and similarly to what shown in Figure 34, the percentage of alternative unknown syllables (here, *EARLY15*, *EARLY30* and *EARLY45*) decreases over time (Figure 36(d-f)).

The fact that the performance obtained for $ld = 3$ is comparable with the performance obtained for $3 > l2 \leq 6$ and better than the performance obtained for $1 \leq ld \leq 2$ (Figure 36) is also confirmed by a better mean spectrogram representation 37. A good epoch of training is determined by looking at the classifier distribution (shown on the top of each spectrogram in Figure 37. The mean spectrograms obtained for $ld = 1$ (Figure 37a) and $ld = 2$ (Figure 37b) show noisy representations for several syllables. For instance, but not restricted to, syllables *C*, *H*, *O*. Nevertheless, some syllable representation are influenced by the fact that the trainings used to generated the samples have been done using the preliminary dataset described in Appendix I.



(a) Number of represented classes.

(b) Median of the number of syllables per class.

(c) Inception score.

(d) Number of syllables belonging to the EARLY15 class.

(e) Number of syllables belonging to the EARLY30 class.

(f) Number of syllables belonging to the EARLY45 class.

Fig. 36. **Comparison between different latent space dimension: quantitative measure.**.Each line represents one instance of training at a particular latent space dimension. Panel (a) shows how many syllables of the repertoire are covered by the generator across time. Panel (b) shows how many syllable per class have been generated in average. The dark lines show the evolution of the mean, whereas the light lines shows the evolution of the median. To build these two panels (a) and (b), $1k$ generations have been generated every 15 epochs, *classifier-EXT* has been used to provide the classification, and only the repertoire's classes have been taken into account when plotting. Panel (c) shows the evolution of the inception score over time. To build this panel $16k$ syllables have been generated every 200, and *classifier-REAL* has been used to provide the classification. Panels (d-f) show the percentage of generated syllables belonging to classes *EARLY15*, *EARLY30* and *EARLY45* across time in comparison with the percentage of syllables belonging to the same class in the training data. A 3-dimensional WaveGAN (orange line) reacheas convergence as good as higher-dimensional WaveGANs(blue, green and magenta lines) and better than lower dimensional WaveGANs (purple and yellow lines). We varied the latent space dimension as $ld = 1, 2, 3, 4, 5, 6$ and we kept fix the training dataset.

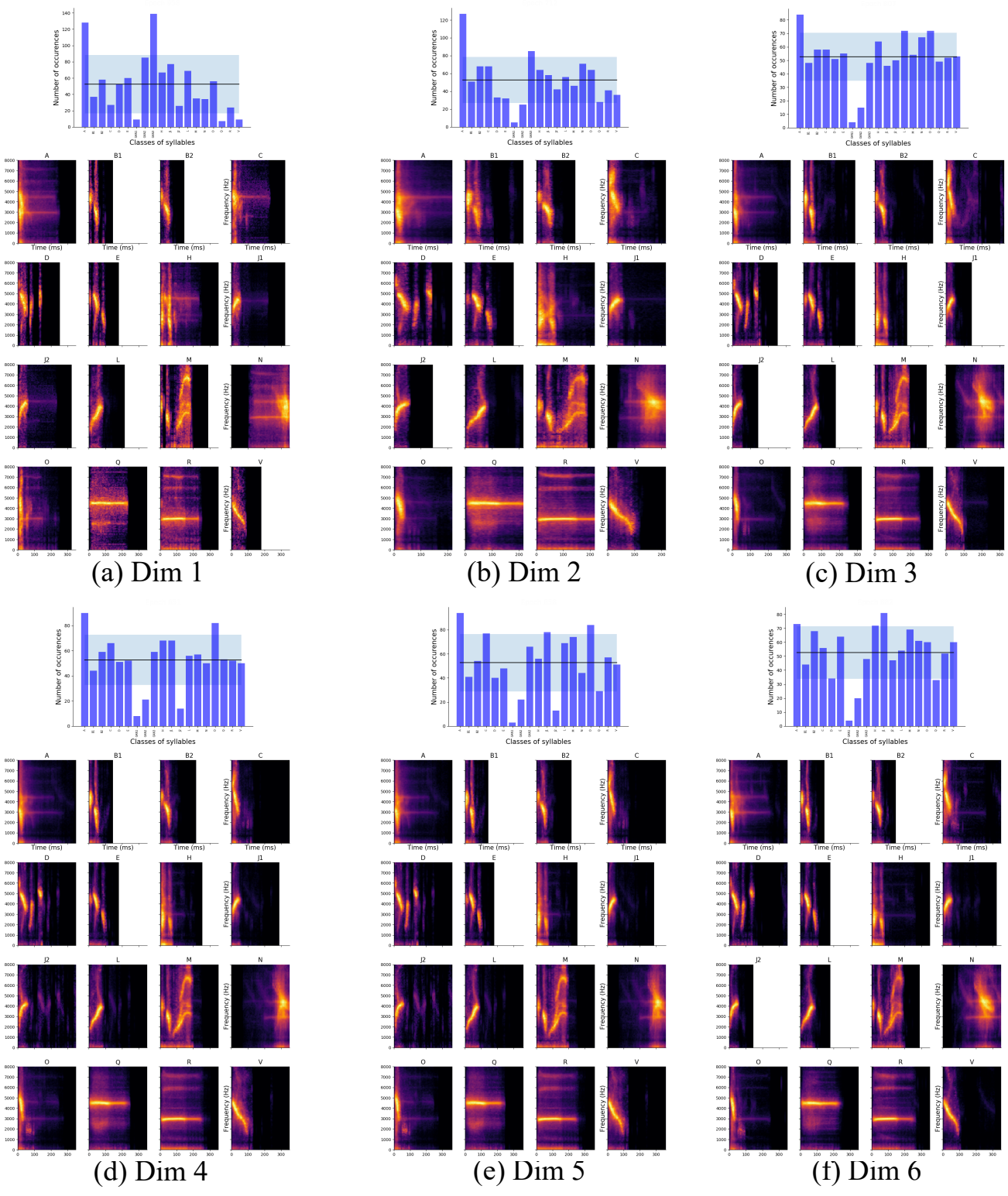
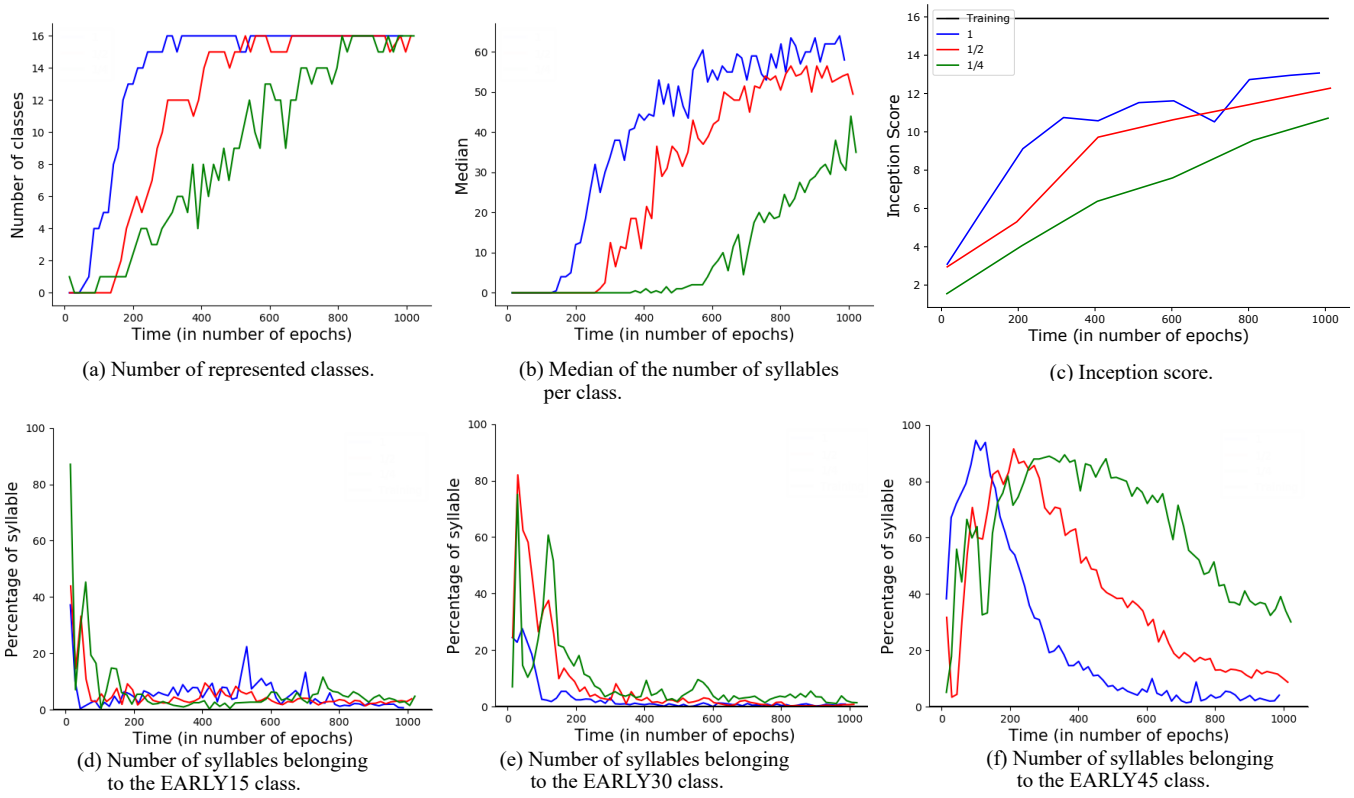


Fig. 37. **Comparison between different latent space dimension: qualitative measure**. Mean spectrogram of $1k$ syllables generated at a good epoch of training determined by looking at the classifier distibution (on top of each spectrogram) for $ld = 1, 2, 3, 4, 5, 6$. Whereas the representations obtained for $ld = 1$ (a) and $ld = 2$ (b) show a noisy representation for several syllables (e.g., syllable *M* in (b)), $ld = 3$ (c) shows a representation comparable to the one obtained for higher conditions (d-f).

2) Dataset size: Complete version of Figure 14 (Figure 38). Accordingly to what observed in the main paper for Figure 36(a-c) and similarly to what shown in Figure 34 and Figure 36, the percentage of alternative unknown syllables (here, *EARLY15*, *EARLY30* and *EARLY45*) decreases over time, and eventually increases when overtraining begins (e.g., the blue line around epoch 900 in Figure 36e).



(a) Number of represented classes.

(b) Median of the number of syllables per class.

(c) Inception score.

(d) Number of syllables belonging to the EARLY15 class.

(e) Number of syllables belonging to the EARLY30 class.

(f) Number of syllables belonging to the EARLY45 class.

Fig. 38. **Comparison between datasets of a different size**. We varied the dataset size as $d = 16000, 8000, 4000$ and we kept fix the latent space dimension at $ld = 3$. Each line in this figure represents one instance of training at a particular dataset size condition. Panel (a) shows how many syllables of the repertoire are covered by the generator across time. Panel (b) shows how many syllable per class have been generated in average. The dark lines show the evolution of the mean, whereas the light lines shows the evolution of the median. To build these two panels (a) and (b), $1k$ generations have been generated every 15 epochs, *classifier-EXT* has been used to provide the classification, and only the repertoire's classes have been taken into account when plotting. Panel (c) shows the evolution of the inception score over time. To build this panel $16k$ syllables have been generated every 200, and *classifier-REAL* has been used to provide the classification. Panels (d-f) show the percentage of generated syllables belonging to the alternative classes *EARLY15*, *EARLY30* and *EARLY45* across time in comparison with the percentage of syllables belonging to each garbage class in the training data. A dataset of bigger size (blue line) allows better and faster convergence than having a dataset of lower sizes (red and green lines).