IDIAP RESEARCH REPORT

LEVERAGING SEQUENTIAL STRUCTURE IN ANIMAL VOCALIZATIONS

Eklavya Sarkar Mathew Magimai-Doss

Idiap-RR-06-2025

JULY 2025

Centre du Parc, Centre du Parc, Rue Marconi 19, CH - 1920 Martigny
T +41 27 721 77 11 F +41 27 721 77 12 info@idiap.ch www.idiap.ch

LEVERAGING SEQUENTIAL STRUCTURE IN ANIMAL VOCALIZATIONS

Eklavya Sarkar *
Idiap Research Institute
Ecole Polytechnique Fédérale de Lausanne
Switzerland
eklavya.sarkar@idiap.ch

Mathew Magimai.-Doss,
Idiap Research Institute
Switzerland
mathew@idiap.ch

1 INTRODUCTION

In previous bioacoustics works (Sarkar & Magimai.-Doss, 2023; 2024; Sarkar et al., 2022; 2025), we averaged each data sample's extracted feature embeddings $\boldsymbol{x} \in \mathbb{R}^{N \times D}$, into a vocalization-level representations, denoted as functional vectors $f_\mu = \mu(\boldsymbol{x}) \in \mathbb{R}^D$ or $\boldsymbol{f}_{\mu\sigma} = \big[\mu(\boldsymbol{x}), \sigma(\boldsymbol{x})\big] \in \mathbb{R}^{2D}$. While these 'stats-pooled' representations have proven very valuable for classification tasks, bandwidth analysis, and model adaptation, they ignore the sequential aspect of animal calls: each vocalization is treated like an unordered bag of frame-level feature embeddings. This completely overlooks the fact that many animal arrange acoustically distinct sub-vocalization units in a specifically ordered sequences that carry important communicative and syntactic information (Kershenbaum et al., 2016a). The goal of this final chapter is thus to investigate alternate feature representations that can capture the sequential structure within animal vocalizations, and leverage the unutilized temporal information to improve classification performance.

In order to effectively model sub-vocalization unit level sounds, we turn to symbolic speech tokenization. Recent work has shown that discrete audio tokens obtained through vector-quantization of 'continuous' SSL feature embeddings can effectively encode acoustic information, and thus be utilized for many speech and audio tasks (Guo et al., 2025). Based on this prior, we extend this framework to bioacoustics, and explore whether token sequences can also reveal meaningful structure in animal vocalizations and help distinguish call-types or individual callers. A successful framework could even yield an inventory of recurring acoustic sub-vocalization units in animal communication. To the best of our knowledge, this is the first work to explore discrete audio tokens for computational bioacoustics. To that end, we investigate vector quantization (VQ) and gumbel-softmax vector quantization (GVQ) as tokenization methods for capturing the sequential structure in non-human animal vocalizations.

The rest of the chapter is structured as follows. First, Section 2 provides a brief overview of sequences in animal vocalizations. Then, Section 3 presents an in-depth review of representation learning using discrete audio tokens. Section 4 describes our experimental setup, namely the quantizer training protocol, token sequence generation, and post-processing. In Section 5, we conduct the pairwise distance analysis, and in Section 6 we benchmark the downstream classification performance. Finally, we conclude with implications and directions for future research in Section 7.

2 SEQUENCES IN ANIMAL VOCALIZATIONS

The communicative power of sequences in animal vocalizations is well-documented across species, with vocal sequences often serving key biological roles such as territory defense, mate attraction, social bonding, and alarm signaling (Kershenbaum et al., 2016b). The complexity of these sequences manifests through distinct patterns of acoustic units that are combined in species-specific ways, following implicit or explicit syntactic rules. For instance, songbirds produce vocalizations composed of repeated motifs and notes arranged in recognizable patterns (Catchpole & Slater, 2003), while cetaceans exhibit intricate, temporally-structured acoustic sequences associated with social interaction and individual identification (Mercado & Handel, 2012). Thus, capturing and analyzing the

*eklavyafcb.github.io

inherent sequential structure in animal vocalizations could substantially enhance our understanding of their communicative function and biological significance.

Several approaches have been proposed in the biological literature to analyze the temporal and structural complexity of vocalizations. These include methods derived from information theory and Markovian analyses of transitions between acoustic units (McCowan et al., 1999), as well as pattern recognition techniques applied directly to acoustic sequences (Kershenbaum et al., 2012). However, these biologically-driven studies often rely heavily on manual annotations or simple acoustic measurements, limiting their scalability and computational generality.

3 Discrete Audio Tokens-based Representation Learning

Many self-supervised speech SSL models, including those we have utilized throughout this thesis, employ discrete token representations during their pre-training stages. Typically, these discrete tokens are derived using a quantization process, either through integrated Vector Quantization (VQ) layers (Baevski et al., 2020a;b) or offline clustering mechanisms applied to continuous embeddings (Hsu et al., 2021). However, such discrete representations are primarily intended to facilitate self-supervised learning objectives, such as masked prediction or contrastive learning, and are usually not directly exposed or utilized during inference or downstream tasks.

In this chapter, we explicitly leverage the discrete tokenization methodology. To do so, we first extract window-level embeddings from a pre-trained SSL model, consistent with our earlier experiments, and subsequently train a separate quantization module which maps the embeddings into sequences of discrete tokens. Note that the quantization is performed independently per frame, thereby preserving the temporal order of the original acoustic events within the vocalization, and is trained separately on extracted embeddings from the pre-trained encoder, using the bioacoustic data of interest. This allows the codebook vectors to adapt specifically to the acoustic characteristics and distributions of the vocalizations being studied.

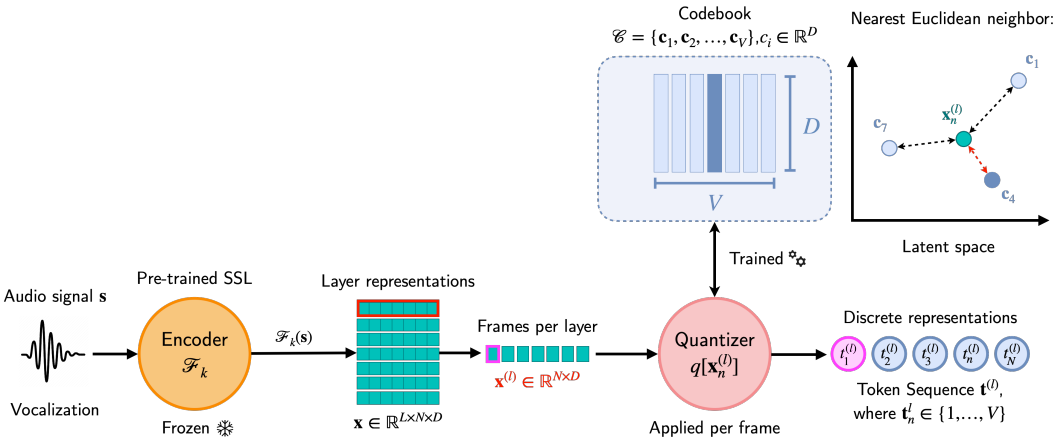

Figure 1: Discrete call tokenization pipeline using vector quantization.

The overall call tokenization pipeline, employed in this work, using vector quantization is illustrated in Figure 1. Specifically, a raw audio waveform $\boldsymbol{s}$ is first passed through a pre-trained encoder $\mathcal{F}$, producing continuous layer embeddings $\boldsymbol{x} \in \mathbb{R}^{L \times N \times D}$, where L is the number of layers, N the number of frames in each layer, and D the dimension of each frame. Let $\boldsymbol{x}_n^{(l)} \in \mathbb{R}^D$ denote the embedding extracted from encoder layer l at frame position n. Each layer embedding is then quantized individually per-frame by a quantization function q, resulting in discrete tokens $\boldsymbol{t}_n^{(l)} = q[\boldsymbol{x}_n^{(l)}]$. Formally, the quantization function maps each embedding from continuous D-dimensional space to a discrete integer token index $q : \mathbb{R}^D \rightarrow \{1, 2, \ldots, V\}$ where V denotes the vocabulary size, i.e., the number of unique discrete tokens. Each token index corresponds directly to an entry in a finite set, referred to as the codebook $\mathcal{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_V\}$, where each code-vector $\boldsymbol{c}_i \in \mathbb{R}^D$ corresponds to the i-th discrete token in the original embedding space. This discretization step

effectively compresses the representation since encoding tokens only requires $\lceil \log_2 V \rceil$ bits per frame.

Detailed descriptions of vector quantization and Gumbel-Softmax vector quantization, which are the specific methods employed to train these quantization modules, are provided in Section 3.1 and Section 3.2, respectively. We specifically leverage them due to their proven effectiveness in quantizing audio embeddings. Note that these are both examples of single-codebook quantizers. Most modern acoustic tokenizers have multiple quantizers. However, for simplicity and clarity, we focus on hand-coded single-codebook ones in this work.

3.1 VECTOR QUANTIZATION (VQ)

While traditional clustering methods operate independently of model training, vector quantization integrates a discrete, learnable codebook directly into the neural network (van Den Oord et al., 2017), enabling end-to-end optimization via gradient propagation through the quantization step.

We maintain a learnable codebook $\mathcal{C} = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_V\} \in \mathbb{R}^{V \times D}$ of $V = 50$ code-vectors, each of dimension $D = 768$. Given an input embedding $\boldsymbol{x}_n^{(l)}$, the quantization process selects the nearest codebook vector $\boldsymbol{c}_i$ by simply minimizing the Euclidean distance between the two:

$$q[\boldsymbol{x}_n^{(l)}] = \underset{i \in \{1,2,\ldots,V\}}{\arg\min} \|\boldsymbol{x}_n^{(l)} - \boldsymbol{c}_i\|_2^2 \tag{1}$$

which returns the token index which is the input's discrete token. The codebook vector itself, which we denote as $\boldsymbol{c}_k \triangleq \boldsymbol{c}_{q(\boldsymbol{x})}$, is passed on to subsequent networks.

To allow backpropagation through the non-differentiable nearest-neighbor argmin lookup given in 1, a *straight-through estimator* (STE) (Bengio et al., 2013) is employed to graft gradients from the quantized output $\boldsymbol{c}_k$ back to $\boldsymbol{x}_n^{(l)}$ during the backward pass. The encoder thus receives learning signals from downstream losses, while the codebook vectors themselves are updated via the VQ loss below. In our case, since we have pre-extracted embeddings, no encoder is updated, and the downstream losses encourage the extracted representations to align with their assigned code-vectors, even though only the codebook parameters are updated. During training, we optimize the VQ loss which is jointly defined as the sum of the codebook and commitment losses:

$$\mathcal{L}_{\text{VQ}} = \underbrace{\|\text{sg}[\boldsymbol{x}_n^{(l)}] - \boldsymbol{c}_k\|_2^2}_{\text{Codebook Loss}} + \underbrace{\beta \|\boldsymbol{x}_n^{(l)} - \text{sg}[\boldsymbol{c}_k]\|_2^2}_{\text{Commitment Loss}}. \tag{2}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator and the beta coefficient is typically set to $\beta = 0.25$. The codebook loss shifts the selected code-vector $\boldsymbol{c}_k$ toward its corresponding input embedding $\boldsymbol{x}_n^{(l)}$, whereas the commitment loss conversely encourages the embedding to move closer to its matched codeword. We iterate $\mathcal{L}_{\text{VQ}}$ over all the layers L and frames N to compute the total cost. While one can also update the codebook via an exponential-moving-average (EMA) scheme (van Den Oord et al., 2017), we focus here on the loss-based updates for clarity. Since the encoder is kept frozen, both terms in practice serve to adapt the codebook vectors to the distribution of the bioacoustic embeddings, yielding a discrete vocabulary that best captures their statistical structure.

VQs are unfortunately also known to suffer from codebook collapse, where the codebook usage is highly imbalanced, i.e. most input embeddings get mapped to a one or two centroids, while the rest of the codebook remains idle and unpdated, drastically reducing its effective representation capacity.

3.2 GUMBEL-SOFTMAX VECTOR QUANTIZATION (GVQ)

To mitigate codebook collapse in the standard VQ, we also implement Gumbel Vector Quantization (GVQ) (Jang et al., 2017), which uses the Gumbel–Softmax relaxation as a proxy for classic Softmax and to enable differentiable sampling from a categorical distribution. Given an input embedding $\boldsymbol{x}_n^{(l)}$, a linear projection layer computes logits $\{\pi_i\}_{i=1}^V$. The relaxed one-hot vector $\boldsymbol{p} \in \Delta^{V-1}$ is then obtained via:

$$p_i = \frac{\exp\big((\log \pi_i + g_i)/\tau\big)}{\sum_{j=1}^{V} \exp\big((\log \pi_j + g_j)/\tau\big)}, \tag{3}$$

where each g_i is an independent sample from the $\mathrm{Gumbel}(0, 1)$ distribution and τ is a fixed temperature (set to 1.0). A straight-through estimator is applied so that, during the forward pass, the highest-probability entry in $\boldsymbol{p}$ is discretized to a one-hot vector, while in the backward pass gradients flow through $\boldsymbol{p}$ as if the operation were identity.

Training of the GVQ module is driven by an entropy-maximizing loss that encourages uniform use of all V codewords. Equivalently, this can be written as a KL divergence between $\boldsymbol{p}$ and the uniform distribution:

$$\mathcal{L}_{\mathrm{GVQ}} = \sum_{i=1}^{V} p_i \log\big(p_i\, V\big) \tag{4}$$

In our GVQ implementation, we implement several extensions to improve codebook utilization and robustness. First, we augment the KL divergence objective with a tunable weight parameter α_{KL}. Second, we add a diversity loss term weighted by a hyperparameter λ_{div}, which explicitly penalizes under-utilization of the codebook. Throughout training, we track two key metrics: the codebook perplexity

$$\mathrm{PPL} \;=\; \exp\Big(-\sum_{i=1}^{V} \bar{p}_i \log \bar{p}_i\Big), \tag{5}$$

where $\bar{p}_i$ is the average probability of selecting codeword i, and the normalized perplexity PPL/V. The diversity loss is defined to increase the normalized perplexity, thereby encouraging the model to make use of a larger fraction of available codewords.

4 Experimental Setup

All of our experiments were conducted using the same preprocessing and batching pipeline to ensure a fair comparison across conditions. For this work, we stuck to HuBERT as our SSL model for extracting feature embeddings $\boldsymbol{x} \in \mathbb{R}^{L \times N \times D}$.

The remaining of this section is organized as follows: Section 4.1 gives an outline of the quantizer training protocol, and Section 4.2 provides the overview of the acoustic token generation, including the sequence post-processing.

4.1 Quantizer Training Protocol

We train all of our vector-quantization models on $\boldsymbol{x}$ using the Adam optimizer with a fixed batch size of 32, running for up to 20 epochs on *Train*, and evaluating performance on a held-out *Val* set to monitor convergence and guard against overfitting. To find the best hyperparameter settings, we conduct a grid search over two quantizer variants, as given in Table 1.

Note that for both quantizer models, the codebook $\mathcal{C}$ is *shared* across all layers L during training. Having the same symbol inventory for every layer makes the token sequences directly comparable across layers, and removes the need to have 13 separate vocabulary sets. Since the codebook must cover the union of all layer manifolds, a codebook-collapse is unlikely, and much less so than the alternate scenario of layer-specific sub-codebooks.

Each mini-batch therefore contains all layers of every utterance during training: batch tensors of shape (B, L, N, D), corresponding to the batch size, layer index, frame index, and feature dimension respectively, are reshaped to $(B \times L, N, D)$, quantized with a $V = 50$ entry codebook, and then reshaped back. This allows the quantizer q to see inputs from all layers, but then generate token sequences $\boldsymbol{t}$ drawn from the common symbol set.

Table 1: Hyperparameter search space for VQ and GVQ models.

Quantizer	Hyperparameter	Search Space
VQ	Learning rate	1e[-4, -3, -2]
	Commitment cost	0.25
	EMA	[True, False]
GVQ	Learning rate	1e[-4, -3, -2]
	KL weight	[0.5, 1.0, 1.5, 2.0]
	Diversity weight	[0.0, 0.01, 0.05, 0.1, 0.2, 0.5]
	Temperature schedule:	
	Max temperature	2.0
	Min temperature	0.1
	Decay factor	0.999

4.2 Token Sequence Generation and Post-Processing

After training the quantizer on *Train*, we generate and save sequences of acoustic discrete tokens $\boldsymbol{t}$ for each vocalization in the entire dataset as described in the pipeline in Section 3. However, during batch processing, audio waveforms are repeat-padded to match the length of the longest sample within the batch. This repetition artificially inflates all the token sequences except one to be longer than the actual audio signals. To account for this, we apply some post-processing to the sequence by first calculating the effective number of frames of each data sample. We determine the downsampling factor of a batch by dividing the longest raw audio length in a given batch by the number of frames in its token sequence. Then, for each data sample, we compute the effective frame count by dividing its raw audio length by this factor and rounding the result. Finally, the token sequence for each sample is trimmed to this effective frame count, yielding a variable-length representation that accurately reflects the original signal duration and excludes any tokens that result solely from the padding. To ensure consistency with the original embedding extraction process, we implement verification mechanisms that confirm sample ordering is maintained throughout the token generation pipeline.

5 Distance Analysis

This section presents a distance analysis for the token sequences to identify any discernible patterns or correlations once we obtain the token sequences for each vocalization using the trained quantizers. Specifically, we are interested in observing the intra-class and inter-class variability to understand the degree with which the generated token sequences are able to distinguish from one class to another.

We use the Levenshtein distance $d(\boldsymbol{t}_1, \boldsymbol{t}_2)$, a string metric also known as the edit distance, to quantitatively measure the distance between a pair of discrete token sequences $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$. The distance effectively represents the minimum number of 'edits', i.e. insertions, deletions, or substitutions, needed to change one sequence into the other. A distance $d = 0$ thus means that the two sequences are identical. It can go up to at most the length of the longer string. However, this metric gives an absolute difference between sequences and is misrepresentative when a pair of sequences have a large difference in lengths. To overcome this issue, we use the normalized Levenshtein distance, which divides the calculated distance by the length of the longer sequence $\frac{d(\boldsymbol{t}_1,\boldsymbol{t}_2)}{\max(|\boldsymbol{t}_1|,|\boldsymbol{t}_2|)}$, where $|\cdot|$ denotes the length of the sequence. In this case, the distance is bounded between 0 and 1, representing identical and completely different sequences respectively. In the case of $d = 1$, one need to edit every character in the longer string to transform it into the other.

We compute the pairwise Levenshtein distances for all data samples, grouping each comparison into one of the following four possible permutations:

(i ●) *Intra-caller, intra-calltype*: two vocalization samples from the same caller producing the same call-type. The distance between these is expected be the smallest.

(ii ●) *Intra-caller, inter-calltype*: two vocalization samples from the same caller producing different call-type.

(iii ●) *Inter-caller, intra-calltype*: two vocalizations from different callers producing the same call-type.

(iv ●) *Inter-caller, inter-calltype*: two vocalizations from different callers producing different call-types. The distance between these is expected be the largest.

Figure 2 presents the means of the distances distributions of the four aforementioned categories, using the token sequences generated from the VQ model. We can observe that groups (i ●) and (iv ●) behave as expected: they both have the smallest and largest distance, on average, for all datasets. We also noticeably observe that group (ii ●)'s distance is larger than group (iii ●)'s for most datasets. This makes sense intuitively: two vocalizations produced by the a caller vocalizing different call-types are more likely to be acoustically distinct, than two generated by different callers vocalizing the same call-type. The discrete acoustic tokens sequences reflect this distribution, demonstrating their ability to model and capture the temporal information encoded in vocalizations.

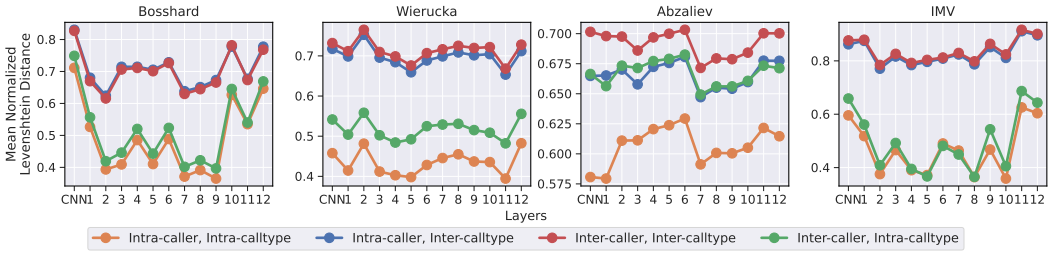

Figure 2: Layer-wise mean Levenshtein distance between all pairs of VQ token sequences.

While we can observe similar trends with the GVQ tokens on the Bosshard dataset, as shown on Figure 3, the remaining datasets exhibit some different patterns. Notably, group (ii ●) and (iii ●)'s trends are flipped in the Abzaliev dataset, showing that the intra-caller, inter-calltype distances are smaller than inter-caller, intra-calltype ones. This may be due to the comparatively large number of callers (80), which increases acoustic variability and makes it harder to distinguish sequences of the same call-type produced by different callers than those of different call-types produced by the same caller. Additionally, for Wierucka and IMV datasets, the pairwise distances in group (iii ●) are unexpectedly smaller on average than in group (i ●). This suggests that the GVQ tokens do not consistently preserve fine-grained caller-specific information as well as the VQ tokens across all datasets.

Taken together, this analysis indicates that the standard VQ discrete token representations are indeed capable of clustering sufficient acoustic information to discriminate by call-type or by caller identity, under real-world, left-to-right temporal constraints. The degree of separability can be measured with a token sequence classification task. The GVQ tokens, however, exhibit some unexpected patterns and less consistent separability, indicating that they may be less effective for classification.

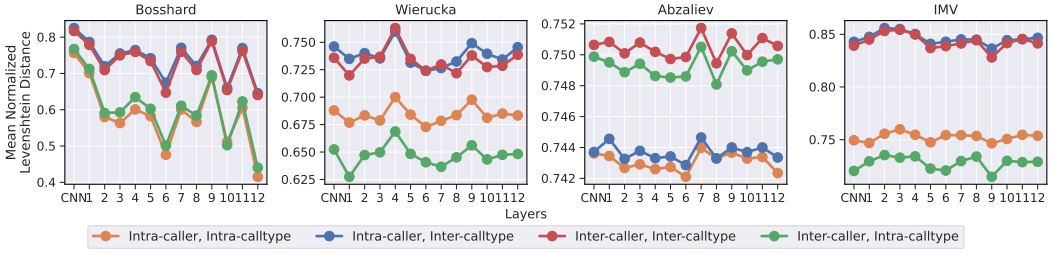

Figure 3: Layer-wise mean Levenshtein distance between all pairs of GVQ token sequences.

6 CLASSIFICATION ANALYSIS

Based on the insights of the comparative analysis, in section, we evaluate how well the sequential nature of token representations can be leveraged for call-type (CTID) and caller (CLID) classification.

6.1 EXPERIMENTAL SETUP

We classify the token sequences using the k-Nearest Neighbours (k-NN) algorithm. We use the pre-computed pairwise Levenshtein distances as our distance similarity matrix, and iterate over the hyperparameters given in Table 2, for each layer, to obtain optimal classification results. The classifier is trained over *Train*, and the hyperparameters defined in the search space are evaluted over *Val*, using UAR as the optimization criterion. The best hyperparameters are then used on *Test*. The predicted label of a sample is determined by applying a majority-voting framework on the actual labels of the k most similar sequences.

Table 2: Hyperparameter search space used for training the k-NN classifier.

Classifier	**Hyperparameter**	**Search Space**
k-NN	Number of neighbours k	[1, 3, 5, 7, 9]
	Neighbour weighting	[Uniform, distance]
	Distance	Levenshtein
	Task	[CTID, CLID]

6.2 RESULTS AND DISCUSSION

The CTID results are shown in Figure 4 for the VQ and GVQ token sequences. We compare the results to a neural linear-probing baseline, as employed in the previous chapters, i.e. by pooling the temporal information into a functional vector $\boldsymbol{f}_{\sigma\mu} \in \mathbb{R}^{2D}$, and classifying it using a fully-connected layer.

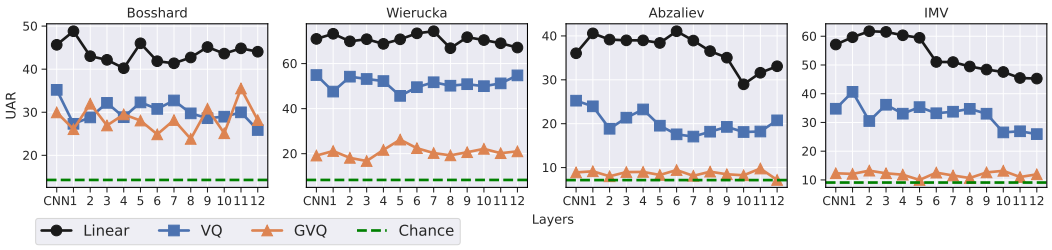

Figure 4: Layer-wise UAR [%] for CTID using k-NN on token sequences

We can observe that the linear layer clearly outperforms the k-NN-based classification of token sequences across all four datasets. Surprisingly, the VQ representations also consistently and substantially perform better than the GVQ ones for all datasets except Bosshard. The same trend can again observed for the CLID task, shown on Figure 5. GVQ especially struggles on the Abzaliev dataset, essentially achieving chance-level performance. This strongly suggests that the GVQ codebook has converged to a local optimum, or potentially collapsed to only a small subset of symbols. In addition, while a single shared codebook can still encode enough information for call–type discrimination, it is perhaps not expressive enough to preserve the finer caller-specific nuances that exist in the continuous embeddings.

The overall trends clearly indicate that while discrete token sequences do carry phonotactic information that can be leveraged, the HuBERT-based feature embeddings still capture much more meaningful information, even when stats-pooled into a vocalization-level vector. In other words, the data tokenization process of converting the feature embeddings causes a higher loss of information

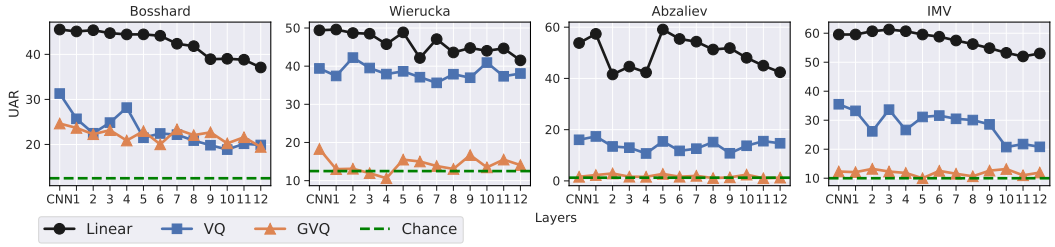

Figure 5: Layer-wise UAR [%] for CLID using k-NN on token sequences.

than what is gained by keeping and leveraging the temporal structure of vocalizations at token-level representations.

Although we trained a single codebook, shared across all layers, for both VQ and GVQ, we still observe that earlier layers tend to yield better performance across tasks, consistent with the trends reported in previous chapters. This indicates that differences between layers persist even after discretization, and that sharing a codebook does not diminish the higher capability of earlier layers in encoding salient and transferable representations.

Table 3: Best UAR [%] scores for each feature across layers. n_C is the number of classes for that dataset and task, and chance performance is calculated as $100/n_c$. $\mathbf{\Delta}$ represents the relative drop in performance with respect to the linear layer baseline.

Task	Dataset	n_C	Chance	Linear	VQ	GVQ	ΔVQ	ΔGVQ
CTID	Bosshard	7	14.30	48.81	35.20	35.52	27.88	27.23
	Wierucka	12	8.30	74.36	54.91	26.23	26.16	64.72
	Abzaliev	14	7.14	41.07	25.24	9.78	38.54	76.20
	IMV	11	9.10	61.75	40.65	24.94	34.17	59.60
CLID	Bosshard	8	12.50	45.52	31.31	24.65	31.22	45.85
	Wierucka	8	12.50	49.60	42.24	18.29	14.83	63.13
	Abzaliev	80	1.25	59.09	17.35	2.90	70.64	95.09
	IMV	10	10.00	61.28	35.51	13.23	42.05	78.42

Table 3 tabulates the highest scores of each feature across layers, and also shows the drop in performance, denoted with Δ, of the token sequence-based representations compared to the linear baseline. Similar to the results in previous chapters, we can see that the CTID classification yields higher scores than CLID across all feature representations. This highlights that call-types differ in distinct spectro-temporal patterns that token sequences can still capture, where as caller identity is largely carried by subtler characteristics that are harder to preserve after vector quantization. This also suggests that discrete tokens need a higher-resolution to be effective.

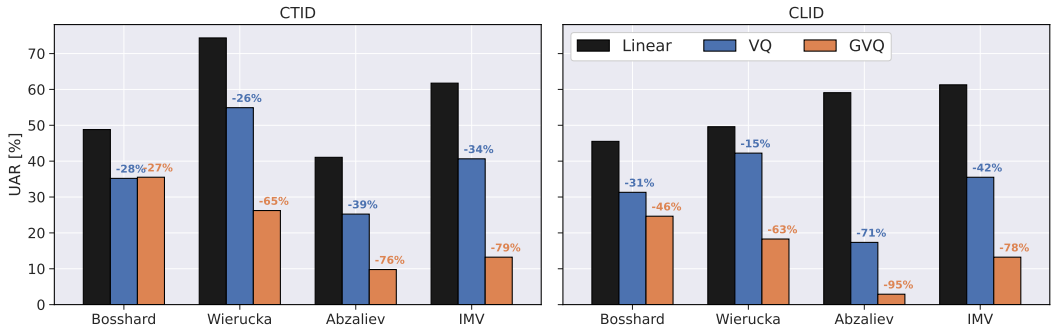

Figure 6: Best UAR results across layers for CTID and CLID.

Figure 6 visually plots the same information. For CTID, discretizing the feature embeddings with a VQ and GVQ drops the performance across datasets by $\sim$26-39% and $\sim$27-79% respectively, when compared to stats-pooling the same features and then classifying with a linear layer. For CLID, the drop is of $\sim$15-71% and $\sim$46-95% respectively. These strong decreases in performances reveal that perhaps a single VQ or GVQ codebook is not enough to effectively model the entire animal vocalizations alone, especially for CLID, or the arbitrary codebook size of $V = 50$. In our early ablation experiments, however, we did not empirically observe a significant change in performance when compared to $V = 25$ or 100. A plausible next step could thus be to train a quantizer model which employs *multiple* codebooks to retain a richer set of temporal patterns.

7 Conclusions and Future Work

In this chapter, we explored alternate feature representations that could preserve the temporal structure of animal vocalizations instead of averaging their extracted SSL feature embeddings into single functional vectors, as in previous chapters. To that end, we investigated and evaluated whether discrete acoustic token-based feature representations could effectively improve call-type and caller classification performance.

To address this problem, we first trained a conventional vector quantization and a Gumbel-softmax vector quantization module to convert the vocalization signals into discrete token sequences for four different animal datasets. In our initial line of investigation, we conducted a comparative analysis of the generated sequences using the Levenshtein distance metric. The results showed that they do encode the sequential structure of animal calls, and exhibit a degree of separability by call-type or caller identity across all datasets. We then trained a k-Nearest Neighbour classifier on said representations to evaluate how well they could systematically distinguish vocalizations by call-type and caller identity. The results showed that both representations were significantly weaker than a simple linear-probe baseline for all datasets. While VQ showed a reasonable performance, GVQ yielded poor scores, nearing chance level in many cases. Overall, the results indicate that token sequences do encode meaningful sequential structure, but the information lost during vector quantization outweighs the benefits gained from explicit temporal modeling.

The scope for improvements on this topic is fairly large. A direct line of investigation would be to improve the quantization module to reduce the information loss. Future work should explore larger, multi-codebook quantization architectures, such as Residual VQs (Juang & Gray, 1982) or Grouped VQ (Jégou et al., 2011). Wav2Vec2 notably employs a grouped VQ module with $G = 2$ codebooks of size $V = 320$. Knowing that its feature embeddings gave a similar performance to HuBERT for Marmoset caller detection in (Sarkar & Magimai.-Doss, 2023), evaluating its token sequences against a matched linear-probe baseline could give meaningful insights.

Another direction of future work could explore more sequence post-processing techniques, such as deduplication, i.e. removing consecutive duplicate tokens (Chang et al., 2024), or acoustic byte-pair encoding (BPE) (Gage, 1994). These can further reduce the sequence length and tighten the alignment between tokens and acoustically meaningful sub-units, which could be particularly useful for vocalizations whose acoustic structure changes slowly.

In summary, despite the promise of symbol-based sequence modeling, this chapter confirms that simple stats-pooled functional vectors remain a highly effective representation for bioacoustic classification tasks, even though they don't directly leverage the temporal structure of animal vocalizations.

Acknowledgments

This work was funded by Swiss National Science Foundation's NCCR Evolving Language project (grant no. 51NF40_180888).

References

Alexei Baevski, Steffen Schneider, and Michael Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations*, 2020a.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460, 2020b.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013.

C. K. Catchpole and P. J. B. Slater. *Bird Song: Biological Themes and Variations*. Cambridge University Press, Cambridge, 2003.

Xuankai Chang, Brian Yan, Kwanghee Choi, Jee-Weon Jung, Yichen Lu, Soumi Maiti, Roshan Sharma, Jiatong Shi, Jinchuan Tian, Shinji Watanabe, Yuya Fujita, Takashi Maekaku, Pengcheng Guo, Yao-Fei Cheng, Pavel Denisov, Kohei Saijo, and Hsiu-Hsuan Wang. Exploring speech recognition, translation, and understanding with discrete speech units: A comparative study. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 11481–11485, 2024.

Philip Gage. A new algorithm for data compression. *C Users J.*, 12(2):23–38, February 1994. ISSN 0898-9788.

Yiwei Guo, Zhihan Li, Hankun Wang, Bohan Li, Chongtian Shao, Hanglei Zhang, Chenpeng Du, Xie Chen, Shujie Liu, and Kai Yu. Recent advances in discrete speech tokens: A review, 2025.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Biing-Hwang Juang and A. Gray. Multiple stage vector quantization for speech coding. In *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pp. 597–600, 1982.

Herve Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

A. Kershenbaum, A. Ilany, L. Blaustein, and E. Geffen. Syntactic structure and geographical dialects in the songs of male rock hyraxes. *Proceedings of the Royal Society B: Biological Sciences*, 279: 2974–2981, 2012.

A. Kershenbaum, D.T. Blumstein, M.A. Roch, Ç. Akçay, G. Backus, M.A. Bee, K. Bohn, Y. Cao, G. Carter, C. Cäsar, M. Coen, S.L. DeRuiter, L. Doyle, S. Edelman, R. Ferrer-i Cancho, T.M. Freeberg, E.C. Garland, M. Gustison, H.E. Harley, C. Huetz, M. Hughes, J. Hyland Bruno, A. Ilany, D.Z. Jin, M. Johnson, C. Ju, J. Karnowski, B. Lohr, M.B. Manser, B. McCowan, E. Mercado, P.M. III, Narins, A. Piel, M. Rice, R. Salmi, K. Sasahara, L. Sayigh, Y. Shiu, C. Taylor, E.E. Vallejo, S. Waller, and V. Zamora-Gutierrez. Acoustic sequences in non-human animals: a tutorial review and prospectus. *Biological Reviews*, 91(1):13–52, 2016a.

Arik Kershenbaum, Daniel T. Blumstein, Marie A. Roch, Çaglar Akçay, George Backus, Mark A. Bee, Kirsten Bohn, Yan Cao, Gerald Carter, Cristiane Cäsar, Michael Coen, Stacy Lynn DeRuiter, Laurance Doyle, Shimon Edelman, Ramon Ferrer-i Cancho, Todd M. Freeberg, Ellen Clare Garland, Morgan Gustison, Heidi E. Harley, Chantal Huetz, Melissa Hughes, Janelle Hyland Bruno, Amiyaal Ilany, Dezhe Z. Jin, Michael Johnson, Chenghui Ju, James Karnowski, Bernard Lohr, Marta B. Manser, Brenda McCowan, Eduardo Mercado, Peter M. Narins, Alex Piel, Meg Rice, Roberta Salmi, Kazutoshi Sasahara, Laela Sayigh, Yu Shiu, Charles Taylor, Elena E. Vallejo, Sara Waller, and Veronica Zamora-Gutierrez. Acoustic sequences in non-human animals: a tutorial review and prospectus. *Biological Reviews*, 91(1):13–52, 2016b.

B. McCowan, S. F. Hanser, and L. R. Doyle. Quantitative tools for comparing animal communication systems: information theory applied to bottlenose dolphin whistle repertoires. *Animal Behaviour*, 57:409–419, 1999.

E. I. Mercado and S. Handel. Understanding the structure of humpback whale songs (l). *The Journal of the Acoustical Society of America*, 132:2947–2950, 2012.

Eklavya Sarkar and Mathew Magimai.-Doss. Can self-supervised neural representations pre-trained on human speech distinguish animal callers? In *Proc. of Interspeech*, pp. 1189–1193, 2023.

Eklavya Sarkar and Mathew Magimai.-Doss. On the utility of speech and audio foundation models for marmoset call analysis. In *4th International Workshop on Vocal Interactivity In-and-between Humans, Animals and Robots (VIHAR2024)*, 2024. ISBN 978-2-9562029-3-6.

Eklavya Sarkar, Pavel Korshunov, Laurent Colbois, and Sébastien Marcel. Are gan-based morphs threatening face recognition? In *International Conference on Acoustics, Speech, & Signal Processing*, May 2022.

Eklavya Sarkar, Amir Mohammadi, and Mathew Magimai-Doss. Adaptation of speech and bioacoustics models. *Idiap-RR*, (Idiap-RR-05-2025), July 2025.

Aaron van Den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017.