7. Appendix

7.1. WavCoch Architecture Details

As shown in Figure 1A, the raw waveform (shape: $1 \times 80{,}000$ for 5s of mono audio sampled at 16kHz) is first transformed into the time–frequency domain via a fixed-kernel discrete Fourier transform implemented as a bank of 1D convolutional filters (window size 1,001 samples, hop length 80 samples). The filter weights—the complex sinusoidal basis functions (or Twiddle Factors [1]) of the discrete Fourier transform—slide over the signal to produce a spectral representation with one feature vector every 5ms. Second, each 5ms temporal step of this frequency representation is passed through an 8-layer encoder stack (each layer is a 1D convolution with 512 channels, kernel size 3, stride 1, ReLU nonlinearities), yielding a sequence of 512-dimensional embeddings. Third, these embeddings are then passed through a 13-dimensional LFQ bottleneck [2], which effectively binarizes the representation. We read out the activations of this bottleneck as a 13-bit binary code which can be interpreted as one of $2^{13} = 8,192$ discrete tokens. We determined that 13-bits is the optimal vocabulary size by ablating vocabulary sizes and evaluating out-of-distribution performance on cochleagram reconstruction error and phoneme cluster purity; 12-bit and 14-bit codes yielded inferior performance (see full ablation details in Appendix 7.2). Fourth, the output of the LFQ bottleneck is passed through a decoder stack (each layer is a 1D convolution with 211 channels, kernel size 9, stride 1, ReLU nonlinearities). This decoder output corresponds to the frequencies in the cochleagram representation [3], which the model is supervised to match via L2 error. An auxiliary entropy penalty with a weight of 0.001 is applied at the LFQ bottleneck to encourage diversity, in line with [2]. Thus, for every 5 seconds of audio, WavCoch extracts a sequence of 988 integers in the range [0, 8192) through the LFQ bottleneck, denoted as cochlear tokens, to feed into AuriStream (illustrated in Figure 1B).

7.2. WavCoch Vocabulary Size Ablations

We performed ablations to identify the optimal vocabulary size of the WavCoch model. We trained variants of WavCoch using a vocabulary size of 4,096, 8,192, and 16,384 (12-, 13- and 14-bit codes, respectively) on the LibriSpeech960 dataset [4]. For each of these models, we evaluated the cochleagram reconstruction L2 error and phoneme cluster purity on an out-of-distribution test set (TIMIT test set [5]). Phoneme cluster purity was defined as *purity = (count of most associated phoneme for token i) / (total counts for token i)* providing an intuitive metric for how consistently a given token aligns with a specific phoneme. Figure I shows that a vocabulary size of 8,192 (13-bit code) yields both the lowest reconstruction error and the highest phoneme cluster purity.

7.3. WavCoch Target Representations: Cochleagram vs. Mel Spectrogram

To evaluate the impact of using the biologically-inspired cochleagram representation [6, 3] as the WavCoch prediction target as opposed to the more standard deep learning practice of using a mel-spectrogram, we trained a version of WavCoch using mel-spectrograms (80 mel bins and 5ms temporal bins) as prediction targets. Both cochleagram- and mel-based WavCoch models were trained on the publicly available LibriSpeech960 dataset [4], consisting of 960 hours of speech recordings. Since the L2 reconstruction error is not directly comparable between a cochleagram and a mel-spectrogram, we investigated two proxy measures of representational quality: i) The number of unique codes utilized in the quantized representation ("codebook usage"), and ii) Phoneme cluster purity (defined as *purity = (count of most associated phoneme for token i) / (total counts for token i)*) Both metrics were computed on the out-of-distribution TIMIT test set [5] and are reported in Table I.

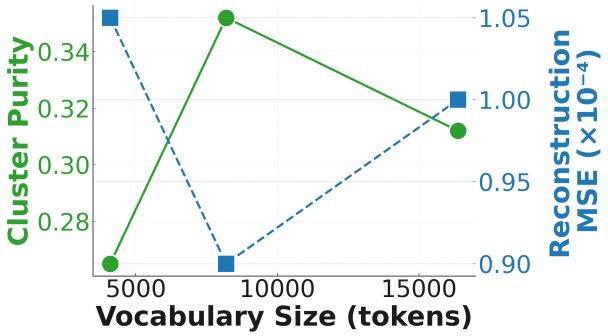

Figure I: ***Evaluation of WavCoch trained with different vocabulary sizes.*** *We plot the L2 cochleagram reconstruction error (blue) and the phoneme cluster purity (green) on the out-of-distribution TIMIT test set.*

Table I: ***Evaluation of WavCoch trained with different prediction targets.*** *Codebook usage and phoneme cluster purity evaluated on the out-of-distribution TIMIT test set.*

Target	Codebook Usage ↑	Cluster Purity ↑
Cochleagram	**8,172**	**0.3517**
Mel-Spectrogram	8,151	0.3473

First, in terms of codebook usage, we found that the WavCoch model trained with the cochleagram target utilized slightly more codes than the model trained with the mel-spectrogram target to represent out-of-distribution speech data (TIMIT test [5]). Second, the cochleagram-based WavCoch model achieved a slightly higher average phoneme cluster purity on the TIMIT test set than the mel-spectrogram model. While these differences are relatively small, they suggest that the cochleagram representation performs at least as well as, if not slightly better than the mel-spectrogram in this setting.

Beyond the quantitative analyses reported in Table I, we prefer the cochleagram over the mel-spectrogram representation for conceptual reasons: The ultimate goal of our framework is to move towards more biologically plausible speech models, and the cochleagram is more aligned with this goal.

7.4. Comparison Models

AuriStream is compared to five state-of-the-art speech representation models using the HuggingFace Transformers package: HuBERT-base (identifier: *facebook/hubert-base-ls960*), HuBERT-xl (identifier: *facebook/hubert-xlarge-ll60k*), wav2vec2-large (identifier: *facebook/wav2vec2-large*), WavLM-base (identifier: *microsoft/wavlm-base*), and WavLM-large (identifier: *microsoft/wavlm-large*). For the SUPERB benchmark, we additionally compare against two smaller models which share some similarity to AuriStream, specifically, APC and vq-wav2vec.

7.5. Confusion Matrix for Phoneme Decoding

Figure II shows the phoneme confusion matrix for AuriStream-1B in the linear decoding task (see Section 3.1). The error patterns were sensible: for instance, "er" was often confused with "r', or "ah" with "ih".

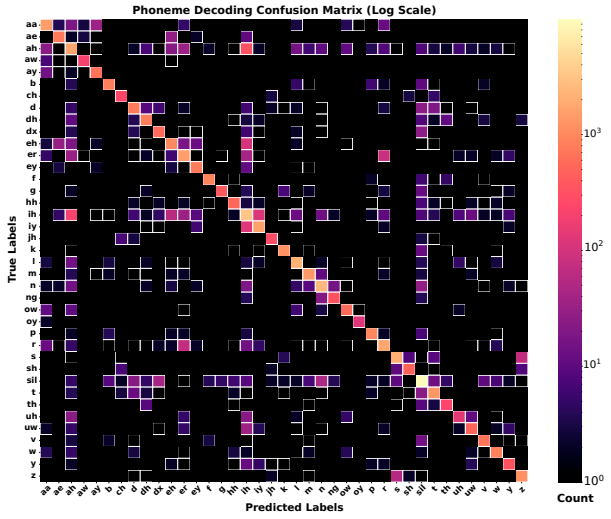

Figure II: ***Confusion matrix for phoneme decoding.*** *The plot shows which phonemes were confused with each other from the AuriStream-1B model on the TIMIT test set. The plot is shown on a log colorscale to better highlight the mismatches between true and predicted labels.*

7.6. Sonifying AuriStream Predictions through Cochleagram Inversion

We investigate AuriStream's predictions by inverting the cochleagrams into audible waveforms. To this end, we developed a simple per-sample optimization procedure that constructs a waveform that matches the cochleagram prediction. Specifically, we optimize a tensor of shape $(1 \times 80{,}000)$—initialized with random numbers from a normal distribution with mean 0 and variance 1—representing the waveform input to make its cochleagram representation match the cochleagram predicted by WavCoch (via L2 error). We backpropagate through the cochleagram transformation and use the Adam optimizer with a learning rate of 1e-2. Note that this optimization procedure is not a learned vocoder model, but a simple procedure which converts the output of WavCoch, the cochleagrams, into audible sound (conceptually similar to Griffin-Lim algorithm).

Several audible samples of speech generations from AuriStream-1B are available at the following link: https://tukoresearch.github.io/auristream-speech/. Please access the page using Google Chrome as we have seen some cases in which Safari and Firefox are not properly loading these videos.

We observed that on short timescales, the model produces reasonable completions, but the longer the completion, the more the predictions drift away from being plausible. We want to emphasize that the purpose of AuriStream is not to be a language model, but a speech representation model—the fact that it can perform rudimentary language modeling is a serendipitous side effect of the training objective, which points to the fact that learning patterns in speech, and producing language may be operationalized under a unified objective. These findings serve as great motivating factors for follow-up work, which will attempt to stabilize speech generations with longer-term coherence, building on the foundation laid out in this paper.

8. References

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of Computation*, vol. 19, pp. 297–301, 1965. [Online]. Available: https://api.semanticscholar.org/CorpusID:121744946

[2] L. Yu, J. Lezama, N. B. Gundavarapu, L. Versari, K. Sohn, D. C. Minnen, Y. Cheng, A. Gupta, X. Gu, A. G. Hauptmann, B. Gong, M.-H. Yang, I. Essa, D. A. Ross, and L. Jiang, "Language model beats diffusion – tokenizer is key to visual generation," 2023.

[3] J. Feather, G. Leclerc, A. Madry, and J. H. McDermott, "Model metamers reveal divergent invariances between biological and artificial neural networks," *Nature Neuroscience*, vol. 26, no. 11, pp. 2017–2034, 2023.

[4] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2015, pp. 5206–5210, iSSN: 2379-190X.

[5] J. S. Garofolo, "Timit acoustic phonetic continuous speech corpus," *Linguistic Data Consortium, 1993*, 1993.

[6] K. Wang and S. Shamma, "Self-normalization and noise-robustness in early auditory representations," *IEEE transactions on speech and audio processing*, vol. 2, no. 3, pp. 421–435, 1994.