# Baseline Report: Conditional Audio Autoencoder

Kirill Shumskiy, Ekaterina Petrova

February 28, 2026

## 1 Baseline Implementation

The baseline model is a class-conditional autoencoder for log-mel spectrograms. Input spectrograms are encoded with a residual convolutional encoder and projected to a latent vector $z$. Class information is injected through learned label embeddings, so both encoding and decoding are conditioned on the target sound class.

The decoder mirrors the encoder with upsampling residual blocks and reconstructs a spectrogram with a tanh output head. Training minimizes reconstruction L1 loss on normalized log-mel features, with AdamW, cosine learning-rate decay, gradient clipping, early stopping, and checkpointing.

In short, this baseline is designed to learn *reconstruction quality first* and then test how far latent sampling can be used for generation.

## 2 Dataset and Preprocessing

Minimal pipeline: compute log-mel spectrograms and save `.npy` features. Defaults: 22,050 Hz, `n_mels=128`, `n_fft=2048`, `hop=512`, fixed `spec_t=176`. In loader, features are pad/crop + normalized to approximately $[-1, 1]$.

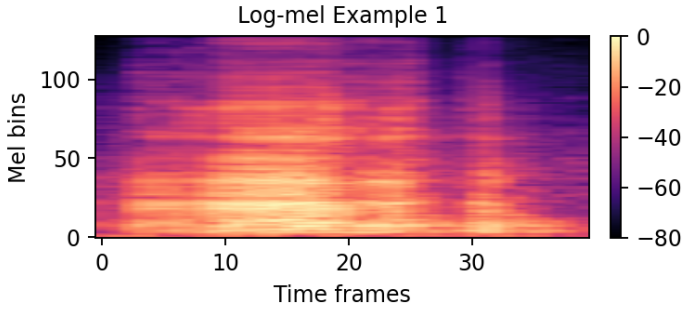


Figure 1: Example of preprocessed log-mel spectrogram used for training.

## 3 Evaluation Outputs

Quantitative metrics: log-mel L1 (primary), MR-STFT, SI-SDR, FAD, diversity.

Qualitative outputs from inference: `spectrograms.png`, `gt.wav`, `recon.wav`, `gen.wav` under run-specific folders.

Executed qualitative export for the best run (`cae_nb_005_..._lat256_lr0.0003`) to: `experiments/cae/cae_nb_005_..` (6 samples). Each sample is stored as `sample_{id}_class_{label}/{spectrograms.png, gt.wav, recon.wav, gen.wav}`.

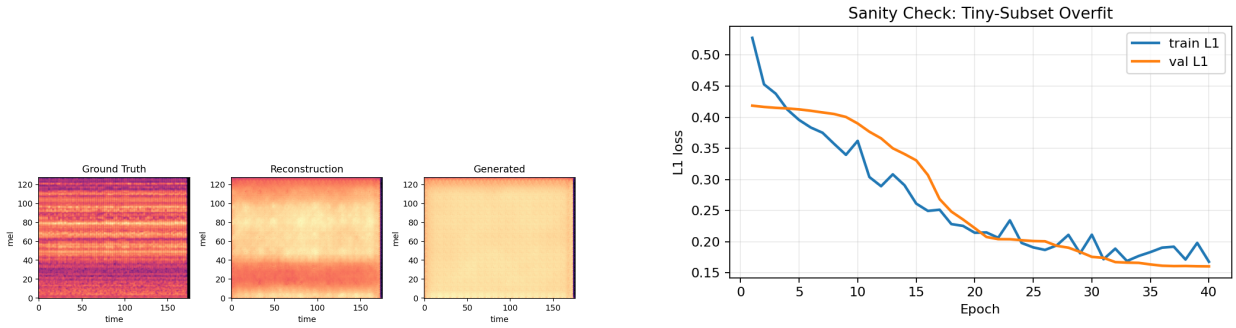


Figure 2: Qualitative sample (left) and sanity-check learning curve (right).

# 4 Conducted Experiments and Results

**Main sweep (6 runs).** Hyperparameters: `latent_dim` $\in$ `{64,128,256}`, `lr` $\in$ `{3e-4,1e-4}`; fixed `batch_size=64`, `spec_t=176`, `n_mels=128`, `n_fft=2048`, `hop=512`. Artifacts are stored in `experiments/cae/`.

| Run setup | Best val L1 | Log-mel L1 | MR-STFT | FAD |
|---|---|---|---|---|
| Lat64, LR 3e-4 | 0.1158 | 0.1158 | 2.1177 | 18077.4 |
| Lat64, LR 1e-4 | 0.1296 | 0.1296 | 2.5739 | 6119.3 |
| Lat128, LR 3e-4 | 0.1099 | 0.1099 | 2.1025 | 31312.7 |
| Lat128, LR 1e-4 | 0.1190 | 0.1190 | 2.3779 | 9289.4 |
| Lat256, LR 3e-4 | **0.1081** | **0.1081** | **2.0958** | 41224.8 |
| Lat256, LR 1e-4 | 0.1094 | 0.1094 | 2.1259 | 14051.2 |

**Sanity check (tiny-subset overfit).** Run: `experiments/sanity_overfit_20260228_214037`. Setup: `overfit_samples=32`, `epochs=40`, `batch_size=16`, `latent_dim=128`, `lr=3e-4`.

Result: best val L1 = 0.1603; train L1 0.5277 $\rightarrow$ 0.1677 ($-68.2\%$), val L1 0.4187 $\rightarrow$ 0.1603 ($-61.7\%$).

# Conclusion

The model works as a *reconstruction* baseline but is weak as a *generator*. In experiments, reconstruction metrics improved (best val/log-mel L1 = 0.1081), while generation metrics stayed poor (high FAD and very low fake diversity versus real diversity). Therefore, the main takeaway is: **this AE does not really learn generation**, which is expected, because a plain AE is not designed to learn a valid sampling prior in latent space.