S-JEPA: SOFT CLUSTERING ANCHORS FOR SELF-SUPERVISED SPEECH REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-supervised speech encoders are predominantly trained by predicting discrete hard cluster IDs at masked positions, a recipe that collapses acoustic ambiguity at category boundaries and requires interrupting training to re-cluster the entire corpus between iterations. We introduce **S-JEPA**, a JEPA-style encoder-predictor pair trained to match the soft posteriors of a Gaussian Mixture Model at masked positions via KL divergence. Training runs as one continuous optimization trajectory in two phases: a fixed GMM over MFCC features, then an *online* GMM over encoder features, with the input layer selected adaptively from a label-free signal, so the re-cluster step runs online rather than offline and the choice of which transformer layer to cluster on is no longer hand-tuned. Under the SUPERB protocol, S-JEPA achieves the lowest WER among evaluated SSL methods below 90M parameters and matches HuBERT-Base on emotion recognition at roughly half its parameter count, with no offline re-clustering and no pretrained teacher. Pre-training data differs across the compared methods, so this is a comparison at matched parameter count rather than matched data or compute. An analysis of the predictor's per-frame entropy on held-out speech reveals a bimodal distribution with a substantial minority of frames near the entropy of a perfect two-cluster tie, showing that the objective retains inter-cluster mass on frames where a hard target would have to discard it.

1 INTRODUCTION

Self-supervised speech representation learning is dominated by a single recipe: cluster acoustic features offline, then train an encoder to predict cluster IDs at masked positions via cross-entropy. HuBERT (Hsu et al., 2021) introduced it with two iterations of k-means (first on MFCC features, then on intermediate transformer features), and WavLM (Chen et al., 2022) extended it with denoising augmentation. The recipe has two awkward properties. First, hard cluster assignment collapses the acoustic ambiguity at category boundaries (Ladefoged & Johnson, 2014; Stevens, 1998): frames at phone boundaries, transitions, and silence/non-silence shifts are forced into one partition. Second, every iteration is a stop-restart job: train the encoder, freeze it, extract features for the entire corpus, run offline k-means, relabel every frame, and restart. For teams pretraining on domain-specific corpora this is expensive and brittle: the cost grows linearly with corpus size, the layer to cluster on must be chosen by hand, and a wrong choice wastes a full iteration.

We introduce **S-JEPA**, a self-supervised speech objective that addresses both points within a single training pass. S-JEPA trains a JEPA-style encoder–predictor pair (LeCun, 2022; Assran et al., 2023) to match the soft posteriors of a Gaussian Mixture Model (GMM) (Dempster et al., 1977; Bishop, 2006) at masked positions, supervised by a single KL divergence loss. Training proceeds in two phases mirroring HuBERT's clustering iterations: a GMM over MFCC features, then a GMM over encoder features. Phase 2 fits its GMM *online* from the EMA encoder's outputs and selects the GMM's input layer adaptively by tracking effective rank, eliminating the offline re-cluster step. With a 51.8M-parameter encoder, S-JEPA reaches **12.10% WER on LibriSpeech test-clean** under the SUPERB ASR protocol (greedy CTC decoding; 8.50% with 4-gram LM rescoring), the lowest of any evaluated SSL method below 90M parameters, and **64.83% accuracy on emotion recognition**, approximately matching HuBERT-Base at 55% of its parameter count. The online updates fold the GMM's cost into the existing training forward pass, removing the full-corpus relabel pass that HuBERT and WavLM use between iterations.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

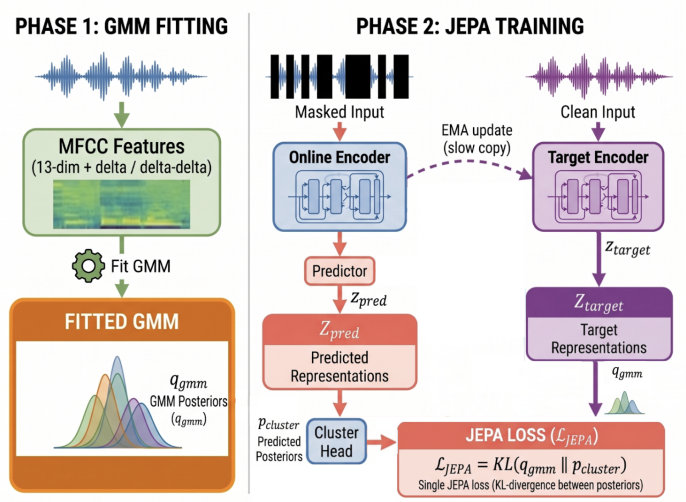


Figure 1: **S-JEPA: a JEPA-style encoder–predictor pair matches soft GMM posteriors at masked positions, with the GMM fit in Phase 1 (frozen, over MFCC features) and updated online in Phase 2 (over EMA encoder features).** The single training signal is KL divergence at masked positions; the predictor, cluster head, and GMM are discarded after pre-training.

The soft-target objective is also analytically informative. On 153K held-out frames, the predictor's per-frame entropy is bimodal: a confident regime concentrated below 0.3 bits, and a clear secondary mode near 1 bit corresponding to two-way ties between competing clusters. Over a third of frames carry more than 1 bit of entropy. These are frames where a hard-target objective would have to commit to one cluster ID and discard the inter-cluster mass. This describes the trained encoder, not where in the signal the uncertainty sits — the measurement is over all frames, not conditioned on boundaries. It characterizes *what* the encoder learned without us having to claim it explains *why* the method is parameter-efficient.

Contributions.

- **A soft-target masked-prediction objective for speech SSL.** HuBERT's hard k-means labels and cross-entropy loss are replaced by the soft posteriors of a GMM over acoustic features, matched by KL divergence at masked positions — a single loss, architecturally compatible with the JEPA pattern (cf. Section 3).
- **A single-pass pipeline that keeps the target soft.** Online clustering has already been used to remove the offline re-cluster step (Liu et al., 2023); what is new here is doing it with a probabilistic mixture, so the target remains a posterior rather than a codebook index. Phase 2 fits its GMM from minibatch sufficient statistics, picks its input layer by effective rank instead of clustering every layer, and alternates the target-encoder EMA decay between two values, giving one continuous optimization trajectory (cf. Section 3.3).
- **The strongest sub-90M encoder among the SSL methods we evaluate, with structured predictor uncertainty.** Trained directly at 51.8M parameters in one pass, with no pre-trained teacher and no offline re-clustering, S-JEPA leads every SSL method below 90M on ASR and emotion recognition, is essentially tied with the best on slot filling, and matches HuBERT-Base on emotion recognition at 55% of its parameter count (cf. Section 4). The predictor's per-frame entropy on held-out speech is bimodal, a substantial minority of frames falling in a two-way-tie regime a hard target could not represent (cf. Section 5).

2 Related Work

Speech SSL families. Self-supervised speech encoders fall into two broad families. *Contrastive* methods predict future or masked latents and contrast against negatives: CPC (van den Oord et al.,

2018), wav2vec (Schneider et al., 2019), and wav2vec 2.0 (Baevski et al., 2020b), which quantizes the latent space. *Masked-prediction* methods reframe SSL as classification over offline cluster IDs at masked positions: HuBERT (Hsu et al., 2021) clusters MFCC features and then intermediate transformer features via offline k-means and supervises with cross-entropy; WavLM (Chen et al., 2022) adds denoising augmentation and a gated relative position bias; w2v-BERT (Chung et al., 2021) combines contrastive and masked-prediction losses. A separate line of work avoids explicit target discretization with EMA teachers (Grill et al., 2020; Caron et al., 2021; Baevski et al., 2022), supervising via regression on continuous teacher representations. JEPAs (LeCun, 2022; Assran et al., 2023; Fei et al., 2024) predict representations across views in a learned latent space. S-JEPA follows the JEPA architectural pattern (encoder, mask token, predictor) and the masked-prediction framing, but supervises with soft categorical targets rather than hard cluster IDs or continuous regression targets.

Clustering methods. Two offline-clustering methods clarify what S-JEPA does and does not do. HuBERT shares its categorical target structure and two-iteration cardinality schedule, but assigns each frame to one cluster by hard k-means and refits centers offline between iterations; S-JEPA replaces both, with soft GMM posteriors on the target side and online updates on the optimization side. BEST-RQ (Chiu et al., 2022) meets the same re-cluster cost by freezing a random-projection quantizer for the whole run, removing the iteration rather than its cost; S-JEPA keeps the iterative structure and lets it run continuously from the encoder's own evolving features.

Online clustering. DinoSR (Liu et al., 2023) is the closest prior work to our Phase 2 and, like S-JEPA, removes the offline re-cluster step. It maintains a codebook of centroids over an EMA teacher's representations, assigns each frame to its nearest centroid under Euclidean distance, and updates the codebook by an exponential moving average of the summed and counted assignments — online k-means. Two differences are load-bearing. First, DinoSR's assignment is hard: the student is trained by cross-entropy against a one-hot centroid index, so the inter-cluster mass that Section 5.2 measures is discarded. Running the clustering online removes the stop-restart pipeline; it does not make the target soft. Our GMM instead supplies a full posterior, with mixture weights and per-component diagonal covariances, matched by KL divergence. Second, because the best layer to cluster drifts as the teacher evolves, DinoSR clusters at the top eight layers at once with a separate codebook for each; we track effective rank and anchor a single GMM at the current argmax. data2vec (Baevski et al., 2022) shares the EMA-teacher and single-pass structure but regresses continuous teacher representations, giving up the categorical target that both DinoSR and S-JEPA rely on. In vision, iBOT (Zhou et al., 2022) and MaskCLIP (Dong et al., 2023) likewise train a masked student against an online EMA teacher, with iBOT's online tokenizer supplying soft distributions over learned prototypes; S-JEPA instead derives its soft targets from an explicit mixture model over acoustic features.

Distillation methods. A separate line of work targets smaller speech encoders by distilling a pretrained model: DistilHuBERT (Chang et al., 2022), FitHuBERT, and LightHuBERT all compress a $\sim$95M-parameter HuBERT teacher into a smaller student. The student's parameter count is the inference-time cost of the compressed model, not the training-time cost of producing it: each of these methods inherits the full HuBERT pretraining pipeline as a prerequisite. S-JEPA reaches the same parameter regime through pretraining alone, with no teacher.

Lineage. Soft cluster supervision via Gaussian mixtures (Dempster et al., 1977; Bishop, 2006) matched by KL divergence (Tishby et al., 2000) is classical. We are not proposing a new clustering method, prediction architecture, or speech encoder backbone; the contribution is the recipe that combines soft GMM posteriors, online updates, and JEPA-style masked prediction into a single-pass speech SSL training pipeline.

3 METHOD

3.1 ARCHITECTURE

The architecture follows the JEPA pattern with three trained components: an encoder f_ϕ mapping the raw waveform to frame-level representations, a predictor h_ψ, and a cluster head g_ω, a shared

MLP mapping frame-level representations to K-dimensional logits. In Phase 2, an *EMA encoder* $\bar{f}$, a frozen copy of f_ϕ whose parameters are an exponential moving average of those of f_ϕ, supplies the input features to the online GMM (cf. Section 3.3); it is unused in Phase 1. The cluster head is applied to the encoder output at visible positions and to the predictor output at masked positions. The masked path always contributes to the loss; the visible path is supervised in Phase 1 and early Phase 2, then dropped.

Concretely, let mask $\in \{0,1\}^T$ be a binary mask marking visible (1) and masked (0) positions. Let $z = f_\phi(x)$ be the encoder output and let $\tilde{z}$ be z with masked positions zeroed out. The predictor injects a learned mask token at the masked positions of $\tilde{z}$, adds positional embeddings, and runs a small Transformer (a single 768-dimensional encoder layer with 8 attention heads) to produce $\hat{z} = h_\psi(\tilde{z}, \text{mask})$. The cluster head gives logits $u_t^{\text{vis}} = g_\omega(z_t)$ at visible positions (supervised in Phase 1 and early Phase 2) and $u_t^{\text{mask}} = g_\omega(\hat{z}_t)$ at masked positions (see Eq. 2). We write $p_t = \text{softmax}(u_t^{\text{mask}})$ for the predictor's softmaxed output at masked frame t, which is what is matched against the GMM posterior in the training loss. After training, only f_ϕ is retained for downstream use; the predictor, cluster head, and GMM are discarded. Full architectural details are in Appendix A.

3.2 Soft Targets and KL Loss

A K-component diagonal-covariance GMM (Dempster et al., 1977; Bishop, 2006) provides soft posteriors over an acoustic feature space:

$$q_k(m) = \frac{\pi_k \mathcal{N}(m; \mu_k, \sigma_k^2)}{\sum_j \pi_j \mathcal{N}(m; \mu_j, \sigma_j^2)}. \tag{1}$$

At each frame t, an acoustic feature vector m_t is computed under `no_grad` (from 39-dimensional MFCC features in Phase 1, or from an EMA encoder in Phase 2; see Section 3.3), and the corresponding posterior $q_t = (q_1(m_t), \ldots, q_K(m_t))$ is used as the soft target. These posteriors assign nonzero probability to multiple components, so frames near a category boundary are not forced into a single partition.

The training loss is the KL divergence between the predictor's softmaxed output and the GMM posterior, averaged over a subset of frame positions S:

$$\mathcal{L} = \frac{1}{|S|} \sum_{t \in S} \text{KL}(q_t \,\|\, p_t)\,, \tag{2}$$

where both arguments are distributions over the same K GMM components. The set S is either the masked positions $\mathcal{M}$, or the union of masked and visible positions, depending on the training stage (cf. Section 3.3).

3.3 Two-Phase Training

Training proceeds in two phases that run as one continuous optimization trajectory.

Phase 1: MFCC GMM. A K=100 diagonal-covariance GMM is fit once over 39-dimensional MFCC features (13 cepstral coefficients with first and second deltas) from the training corpus, initialized with mini-batch k-means (MacQueen, 1967) on a reservoir-sampled subset and refined with expectation-maximization (Dempster et al., 1977). The GMM is then held fixed throughout Phase 1. At each training step, GMM posteriors are computed on the MFCCs of each utterance and used as targets in Eq. 2. Phase 1 applies the loss at both masked and visible positions ($S = \mathcal{M} \cup \mathcal{V}$) with denoising augmentation enabled. Full details are provided in Appendix E.

Phase 2: online encoder-feature GMM. At the Phase 1→2 boundary, the EMA encoder $\bar{f}$ is initialized from the Phase 1 encoder, and a K=500 GMM is initialized over $\bar{f}$'s frame-level features at the active layer using the same mini-batch k-means + EM procedure as in Phase 1. Phase 2 begins with the same loss and augmentation configuration as Phase 1; partway through it transitions to a masked-only loss with augmentation off (cf. Appendix C). K=100 and K=500 were selected to match the cluster cardinalities used in HuBERT iterations 1 and 2 respectively. Since the cardinality

changes, the cluster head is re-initialized at the phase boundary with a 500-way output; the encoder and predictor carry over unchanged. We did not sweep K because the goal is to isolate the effect of soft versus hard targets at matched cardinality, not to optimize K.

Online GMM updates. Updating the GMM online lets it track the encoder's features as they evolve, rather than freezing a snapshot at a re-cluster boundary. After each minibatch we compute responsibility-weighted means and variances of the EMA encoder's frame-level features and EMA-update the GMM parameters with a slow decay; no gradient flows through the GMM. The offline re-cluster used by HuBERT and WavLM costs $O(N_s D + I N_s K D)$ time and $O(N_s D + KD)$ memory, for N_s subsampled frames of dimension D, K clusters and I k-means iterations, plus a full relabel pass over the corpus; for our 83K-hour corpus at 50 Hz, one k-means iteration alone processes $\sim$150M frames. S-JEPA instead pays $O(BKD)$ per step for a minibatch of B frames (192 utterances of up to 15 s, i.e. at most $\sim$144K frames), alongside the existing training forward pass. Update equations are in Appendix E.

Adaptive layer selection. The encoder layer used as the GMM's input is selected adaptively rather than fixed by hand. Let $Z \in \mathbb{R}^{N \times D}$ denote a centered batch of N=2,000 frame embeddings of dimension D from a given encoder layer, with singular values $\sigma_1 \geq \cdots \geq \sigma_D$ and normalized distribution $p_i = \sigma_i / \sum_j \sigma_j$. The *effective rank* of Z is $\mathrm{erank}(Z) = \exp\Big(-\sum_{i=1}^{D} p_i \log p_i\Big)$, the exponential of the Shannon entropy of the normalized singular value spectrum. It is high when many singular values are comparable, low when a few dominate; we use it as a label-free proxy for representational richness. It lower-bounds matrix-based entropy in the Shannon case (Skean et al., 2025, Theorem 1), and spread-of-spectrum measures of this family correlate with downstream performance across transformer and state-space architectures, including as label-free layer-selection signals (Skean et al., 2025; Garrido et al., 2023). At a fixed cadence during Phase 2, we compute $\mathrm{erank}(Z^{(\ell)})$ at each transformer layer on a batch of EMA-encoder features, EMA-smooth across measurement intervals, and select the argmax as the GMM's input layer. When the argmax changes, the GMM continues updating online from the new layer with no offline re-cluster step. We additionally calibrated effective rank against per-layer downstream WER at the end of Phase 1 in our specific setting (cf. Appendix G), a single check at the phase boundary rather than a guarantee that the proxy holds throughout Phase 2.

In our 6-layer encoder (indexes 0-5), the active layer was initialized at $\ell^\star = 2$ (the highest-effective-rank layer index at the end of Phase 1) and the adaptive rule transitioned it to $\ell^\star = 4$ midway through Phase 2, where it remained (cf. Figure 4 in Appendix G). Tracking development-set WER throughout Phase 2, we observed that once layer 2 ceased to be the highest-effective-rank layer, anchoring the GMM at $\ell^\star = 2$ was correlated with stalled WER improvement; switching to $\ell^\star = 4$ was correlated with continued progress. We do not claim higher effective rank causes better downstream performance, only that anchoring at the current argmax is correlated with continued progress.

Periodically switched EMA decay. Self-distillation usually uses a fixed slow EMA decay (e.g., $\alpha = 0.9999$) so the EMA encoder is nearly stationary (Grill et al., 2020; Baevski et al., 2022). That suits continuous regression targets, but here the EMA encoder's outputs are the input features from which the online GMM's components are estimated: held slow throughout, the GMM never sees recent improvements in the online encoder; held fast, the target distribution shifts on every step. We therefore switch the decay rate periodically between two fixed values:

$$\alpha_t \in \{\alpha_{\text{fast}}, \alpha_{\text{slow}}\}, \quad \alpha_{\text{fast}} = 0.999, \quad \alpha_{\text{slow}} = 0.9999, \tag{3}$$

applied to the update $\bar{\phi} \leftarrow \alpha_t \bar{\phi} + (1 - \alpha_t)\phi$. Training begins at α_{fast}, and the rate flips at a fixed step cadence: under α_{fast} the GMM's inputs track recent encoder improvements; under α_{slow} they are held approximately stable so the online encoder can learn against them. Schedule details are in Appendix F.

3.4 Training Algorithm

The full training procedure is summarized as pseudocode in Algorithm 1 (Appendix B). Block masking samples spans of frames until a target fraction is masked. The encoder f_ϕ and predictor h_ψ are

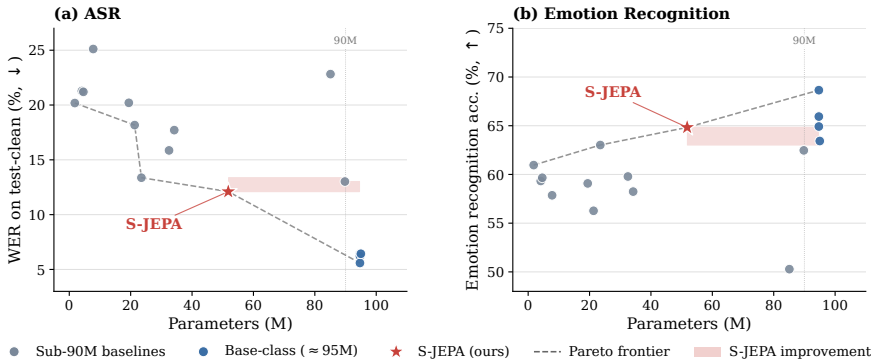


Figure 2: **S-JEPA on the parameter-count frontier on SUPERB below 90M parameters.** Parameter count vs. performance on (a) ASR (WER on LibriSpeech `test-clean`, ↓) and (b) emotion recognition (accuracy, ↑). Dashed line: Pareto frontier across all evaluated SSL methods. Shaded region: S-JEPA's improvement over the prior best in the 51.8M–90M range. S-JEPA reaches 12.10% WER (vs. 13.02% for DeCoAR 2.0 at 89.8M) and 64.83% ER accuracy (matching HuBERT-Base at 55% of its parameters), trained directly without offline re-clustering, hand-tuned layer selection, or a teacher model. The frontier is over parameter count only; pre-training data is not matched across methods.

trained jointly via the gradient of Eq. 2. An EMA copy $\bar{f}$ is initialized from f_ϕ at the Phase 1→2 boundary and maintained throughout Phase 2; its decay rate α_t is periodically switched between α_{fast} and α_{slow} as in Eq. 3. In Phase 1 the GMM is the frozen MFCC GMM and the EMA encoder is unused.

4 Experiments

Setup. We pre-train S-JEPA on a large-scale English corpus combining LibriLight (Kahn et al., 2020) and an English subset of Granary (Koluguri et al., 2025), a total of approximately 83,000 hours, and evaluate it with the encoder frozen on three SUPERB benchmark suite tasks (wen Yang et al., 2021). Pre-training budgets vary widely across the methods we compare: HuBERT, WavLM Base, wav2vec 2.0 and DeCoAR 2.0 use LibriSpeech's 960 hours, while WavLM Base+ uses roughly 94,000. WavLM's training mixture additionally includes proprietary audio not publicly available (Chen et al., 2022), making a fully data-matched comparison infeasible for that baseline. Our comparisons therefore hold parameter count fixed, not data or compute. Following the SUPERB protocol, the encoder is frozen and small task-specific heads are trained on top. The encoder backbone uses 6 Transformer layers (51.8M parameters), targeting the sub-90M parameter regime; full hyperparameters are provided in Appendix D.

Practical advantages over the HuBERT/WavLM pipeline. Beyond parameter count, four properties of S-JEPA matter for deployment. First, no offline re-clustering: training runs as a single script with no stop-restart pipeline, so pretraining on new corpora does not require building feature extraction and k-means infrastructure. Second, adaptive layer selection: the encoder layer used for GMM input is selected during training by effective rank, removing a hand-tuned hyperparameter that, if set incorrectly in HuBERT, wastes a full iteration. Third, no teacher model: unlike DistilHuBERT and FitHuBERT, which require a pretrained 95M teacher as a prerequisite, S-JEPA is trained from scratch. Fourth, smaller inference footprint: at 51.8M parameters, S-JEPA is roughly half the size of HuBERT-Base, reducing inference cost for on-device or edge deployment.

4.1 Frozen-Encoder Results on SUPERB

Table 1 reports frozen-encoder evaluation on ASR (WER on LibriSpeech `test-clean`), ER (emotion recognition accuracy) and SF (slot filling F1 and CER), all under the SUPERB protocol with greedy CTC decoding. S-JEPA reaches 12.10% WER and 3.62% CER; with standard LibriSpeech

Table 1: Frozen-encoder evaluation on the SUPERB benchmark: ASR (WER on LibriSpeech `test-clean`), emotion recognition (ER, accuracy), and slot filling (SF, F1 and CER). All entries use the standard SUPERB protocol with greedy CTC decoding for ASR. Baseline numbers are from the public SUPERB leaderboard. Parameter counts are inference-time encoder only. Best result within the sub-90M block in bold. Selected baselines; the full comparison is in Table 4 (Appendix J).

Method	**# Params**	**ASR** WER (↓)	**ER** Acc. (↑)	**SF** F1 (↑)	**SF** CER (↓)
Sub-90M baselines					
DistilHuBERT (Chang et al., 2022)	23.5M	13.37	63.02	82.57	35.59
wav2vec (Schneider et al., 2019)	32.5M	15.86	59.79	76.37	43.71
vq-wav2vec (Baevski et al., 2020a)	34.2M	17.71	58.24	77.68	41.54
DeCoAR 2.0 (Ling & Liu, 2020)	89.8M	13.02	62.47	**83.28**	34.73
S-JEPA (ours)	**51.8M**	**12.10**	**64.83**	83.05	**33.17**
Base-class (≈95M)					
HuBERT Base (Hsu et al., 2021)	94.7M	6.42	64.92	88.53	25.20
WavLM Base (Chen et al., 2022)	94.7M	6.21	65.94	89.38	22.86

4-gram LM rescoring ($\alpha = 0.5$, $\beta = 1.5$, beam $= 100$), 8.50% WER and 2.90% CER. The greedy figure is the headline, for comparability with the baselines. Every S-JEPA entry is one pre-training run and one head-training run, as on the SUPERB leaderboard the baselines come from; repeating pre-training across seeds was not feasible at this compute scale. Table 2 instead averages over splits and reports standard deviations.

Comparison below 90M parameters. Among the 12 evaluated methods below 90M parameters, S-JEPA leads on ASR and ER and is essentially tied on SF (cf. Figure 2): lowest WER (12.10% vs. DistilHuBERT's 13.37% and DeCoAR 2.0's 13.02%), highest ER accuracy (64.83% vs. DistilHuBERT's 63.02%), and SF F1 essentially tied with DeCoAR 2.0 (83.05 vs. 83.28) at 58% of its parameter count. Unlike DistilHuBERT, whose 23.5M parameters reflect only the student model and require a 94.7M HuBERT teacher to produce, S-JEPA is trained directly at 51.8M in a single pass. Base-class methods (HuBERT-Base 6.42%, WavLM-Base 6.21%, wav2vec 2.0 Base 6.43%) achieve substantially lower WER at roughly twice S-JEPA's parameter count; we claim a frontier within the sub-90M regime, not parity with Base-class methods.

Beyond ASR. On ER, S-JEPA's 64.83% is essentially tied with HuBERT-Base (64.92%) at 55% of its parameters and exceeds every sub-90M baseline (next-best DistilHuBERT, 63.02%). On SF, its 83.05 F1 / 33.17 CER ties DeCoAR 2.0 (83.28 at 89.8M) and beats all evaluated methods below DeCoAR 2.0 in parameter count, while the strongest Base-class methods (WavLM-Base+ 90.58, WavLM-Base 89.38, HuBERT-Base 88.53) lead by 5.5–7.5 F1 — a wider gap than on ASR or ER, consistent with SF rewarding the extra capacity of the 95M encoders.

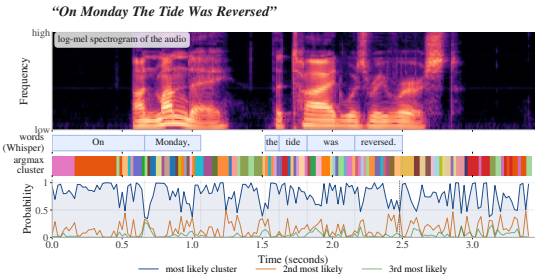
4.2 Probing Frozen Representations

We further probe S-JEPA representations against HuBERT and WavLM on four LibriSpeech (Panayotov et al., 2015) tasks: Speaker Identification, Gender Classification, Chapter Identification, and Phoneme Classification. Chapter Identification is the most discriminative task for this comparison: chapters within a book are often read by the same speaker, so the available cue is subtle prosodic and recording variation rather than speaker identity alone. Probing protocol details (probe architectures, training, splits) are in Appendix H.

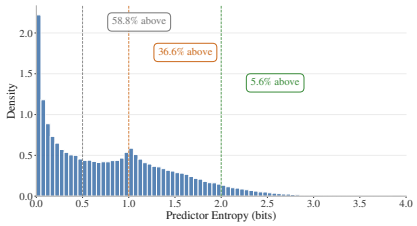
S-JEPA outperforms WavLM and HuBERT on Speaker ID, Gender, and Chapter ID under linear probes (Table 2); the Speaker ID and Gender margins are at the ceiling of the metric, while the Chapter ID gap (S-JEPA 93.2 vs. HuBERT 88.2) is more substantial. On Phoneme classification the linear probe favors WavLM (86.1 vs. 82.8 for S-JEPA), but switching to an MLP probe reverses the ranking (S-JEPA 88.0 vs. WavLM 87.5). The MLP gain over the linear probe is substantially larger

Table 2: **S-JEPA leads on Speaker, Gender, and Chapter ID, and benefits more from a nonlinear probe on phoneme classification.** Probing accuracy on LibriSpeech (Panayotov et al., 2015) across four tasks. Mean ± standard deviation across 3 splits. Chapter ID is the most discriminative comparison: chapters within a book are typically read by the same speaker, so the cue is subtle prosodic and recording variation rather than speaker identity.

Task (Accuracy ↑)	WavLM	HuBERT	S-JEPA
Speaker ID	91.1 ± 1.0	99.1 ± 0.3	**99.7 ± 0.3**
Gender Class.	96.3 ± 0.5	99.5 ± 0.3	**99.6 ± 0.3**
Chapter ID	59.5 ± 1.0	88.2 ± 1.0	**93.2 ± 1.0**
Phoneme (linear)	**86.1 ± 0.2**	84.8 ± 0.1	82.8 ± 0.1
Phoneme (MLP)	87.5 ± 0.1	86.5 ± 0.1	**88.0 ± 0.1**



(a) One held-out utterance. From top: log-mel spectrogram; word-level Whisper alignments; argmax cluster ID per 20 ms frame ($K = 500$); the predictor's top-three cluster posteriors.



(b) Per-frame predictor entropy across ∼153K frames ($N = 500$ `dev-clean` utterances). Reference verticals at 0.5, 1, and 2 bits.

Figure 3: **The soft-target objective produces structured uncertainty, not diffuse fuzziness.** a On a single utterance, rank-1 posteriors sit near 1.0 over sustained acoustic regions and drop to the 0.3–0.45 range near the word boundaries given by the Whisper alignment, with rank-2 rising to compete. b At the population level, measured over all frames rather than boundary frames: a bimodal entropy distribution with a confident regime below 0.3 bits and a clear secondary mode at ~ 1 bit (the entropy of a perfect two-way tie); 36.6% of frames carry more than 1 bit.

for S-JEPA (+5.2 points) than for WavLM (+1.4) or HuBERT (+1.7), suggesting that S-JEPA encodes phonetic information in a form that is more recoverable nonlinearly.

5 Analysis

5.1 Per-utterance Interpretability

On a single held-out LibriSpeech `dev-clean` utterance, masking disabled, we extract from the frozen encoder and cluster head the predictor's distribution p_t over the $K = 500$ Phase-2 GMM components at each 20 ms frame. The time axis is labelled with word-level start/end times from an off-the-shelf small Whisper (Radford et al., 2022) model, purely as a visual reference: these alignments play no role in S-JEPA training or in any quantitative result reported here. The argmax strip in Figure 3a shows cluster IDs turning over several times within a single word. We report this as the rate at which the vocabulary changes, not as evidence of phonetic granularity — the alignments are word-level, and showing the units are finer than phones would need phone-level forced alignment, which we have not run.

5.2 Aggregate Distribution of Predictor Uncertainty

To test whether this generalises beyond one utterance, we sample $N = 500$ `dev-clean` utterances (Panayotov et al., 2015), extract per-frame predictor distributions $\{p_t\}$ with masking disabled,

and convert each frame's entropy to bits, $H_t = -\sum_{k=1}^{K} p_{t,k} \log_2 p_{t,k} \in [0, \log_2 K]$. Bits give K-independent reference points: $H = 1$ is a perfect two-way 50/50 tie, $H = 2$ a four-way uniform spread, $H = \log_2 K = 9$ uniform over all $K = 500$ clusters. The bimodality is the load-bearing observation. Softness that was merely diffuse would give a unimodal distribution sliding from low to high entropy; softness that was a degenerate fallback to uniform priors would pile mass near $\log_2 K$. S-JEPA does neither (cf. Figure 3b): a dominant confident mode concentrated below 0.3 bits, and a clear secondary mode near 1 bit where two clusters tie, with 36.6% of frames above 1 bit. Over a third of frames therefore sit where a hard-target objective would have to commit to one cluster ID and discard the inter-cluster mass. The measurement is over all frames, not boundary frames specifically: Figure 3a suggests the high-entropy frames concentrate near boundaries, but the aggregate does not establish it, and a boundary-conditioned analysis is left to future work.

6 Discussion

S-JEPA can be read as soft-target HuBERT in a JEPA-style architecture: GMM posteriors replace k-means hard labels, retaining the inter-cluster mass a hard assignment discards (Ladefoged & Johnson, 2014; Stevens, 1998), and the predictor-fills-the-mask pattern of JEPA work (LeCun, 2022; Assran et al., 2023; Fei et al., 2024) sits on top of a categorical target. The practical contribution is the pipeline rather than the WER number: the GMM updates online, the clustering layer selects itself by effective rank, and no teacher is needed, so a single training script replaces the stop-restart pipeline of HuBERT and WavLM, and the SUPERB results show this does not sacrifice representation quality.

The paper's two strongest findings sit alongside each other: the sub-90M result of Section 4.1, and the bimodal posterior structure of Section 5.2, where 36.6% of frames occupy a two-way-tie regime a hard target could not represent. That soft categorical targets carry information argmax labels discard is the canonical argument of knowledge distillation (Hinton et al., 2015), recurring in self-distillation with softened teacher distributions over discrete prototypes (Caron et al., 2021). S-JEPA's targets are also discrete — soft posteriors over $K{=}500$ GMM components — and the bimodality suggests the inter-cluster mass separating a soft from a hard target is the information-carrying part of the signal on those frames. A controlled head-to-head ablation against an equivalent hard-target baseline at matched compute is the natural next step.

6.1 Limitations and Future Work

The online GMM, adaptive layer selection and switched EMA decay are evaluated only as a package, not ablated individually. We report one model at one scale on three SUPERB tasks; a data-matched comparison and a scaling study into the Base regime would settle the method's standing. All training and evaluation are in English; tonal languages, low-resource regimes, and non-speech audio remain open.

7 Conclusion

We introduce S-JEPA, a soft-target masked-prediction objective for self-supervised speech representation learning: GMM posteriors at masked positions matched by KL divergence, with the two-phase recipe running as one continuous pass via online GMM updates, adaptive layer selection, and a periodically switched EMA decay. With a 51.8M-parameter encoder, S-JEPA is the lowest-WER SSL method below 90M parameters on LibriSpeech `test-clean` and matches HuBERT-Base on emotion recognition at 55% of its parameters — at matched parameter count, not matched pre-training data. Held-out predictor entropy is bimodal, a substantial minority of frames falling in a two-way-competition regime a hard target could not represent.

8 ACKNOWLEDGMENTS

AI USE STATEMENT

We did not use generative AI tools for any of the tasks for which the ICLR 2027 AI Policy requires disclosure: we did not use them to generate synthetic data, develop theoretical models or conceptual frameworks, propose or refine hypotheses, design or provide feedback on the methodology or experiments, implement methods, clean or reformat datasets, or interpret results; formulating or proving mathematical claims, translation, and qualitative or thematic data analysis are not applicable to this work. We take responsibility for the full content of this work.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *CVPR*, 2023.

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

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *NeurIPS*, 2020b.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In *ICML*, 2022.

Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Heng-Jui Chang, Shu-wen Yang, and Hung-yi Lee. Distilhubert: Speech representation learning by layer-wise distillation of hidden-unit bert. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7087–7091. IEEE, 2022.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. In *IEEE JSTSP*, 2022.

Chung-Cheng Chiu, James Qin, Yu Zhang, Jiahui Yu, and Yonghui Wu. Self-supervised learning with random-projection quantizer for speech recognition. In *ICML*, 2022.

Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass. An unsupervised autoregressive model for speech representation learning. In *INTERSPEECH*, 2019.

Yu-An Chung, Hao Tang, and James Glass. Vector-quantized autoregressive predictive coding. In *INTERSPEECH*, 2020.

Yu-An Chung, Yu Zhang, Wei Han, Chung-Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu. w2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. In *ASRU*, 2021.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B*, 1977.

Xiaoyi Dong, Jianmin Bao, Yinglin Zheng, Ting Zhang, Dongdong Chen, Hao Yang, Ming Zeng, Weiming Zhang, Lu Yuan, Dong Chen, Fang Wen, and Nenghai Yu. MaskCLIP: Masked self-distillation advances contrastive language-image pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Zhengcong Fei, Mingyuan Fan, and Junshi Huang. A-jepa: Joint-embedding predictive architecture can listen. 2024. URL https://arxiv.org/abs/2311.15830.

Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann Lecun. Rankme: Assessing the downstream performance of pretrained self-supervised representations by their rank. In *International conference on machine learning*, pp. 10929–10974. PMLR, 2023.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *ICML*, 2006.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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. In *IEEE/ACM TASLP*, 2021.

Jacob Kahn, Morgane Rivière, Weiyi Zheng, Eugene Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Lber, Steffen Schneider, Emmanuel Dupoux, and Gabriel Synnaeve. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP*, 2020.

Nithin Rao Koluguri, Monica Sekoyan, George Zelenfroynd, Sasha Meister, Shuoyang Ding, Sofia Kostandian, He Huang, Nikolay Karpov, Jagadeesh Balam, Vitaly Lavrukhin, Yifan Peng, Sara Papi, Marco Gaido, Alessio Brutti, and Boris Ginsburg. Granary: Speech recognition and translation dataset in 25 european languages, 2025. URL https://arxiv.org/abs/2505.13404.

Peter Ladefoged and Keith Johnson. *A Course in Phonetics*. Cengage Learning, 7th edition, 2014.

Yann LeCun. A path towards autonomous machine intelligence. *OpenReview*, 2022.

Shaoshi Ling and Yuzong Liu. Decoar 2.0: Deep contextualized acoustic representations with vector quantization. *arXiv preprint arXiv:2012.06659*, 2020.

Alexander H. Liu, Yu-An Chung, and James Glass. Non-autoregressive predictive coding for learning speech representations from local dependencies. 2020a. URL https://arxiv.org/abs/2011.00406.

Alexander H. Liu, Heng-Jui Chang, Michael Auli, Wei-Ning Hsu, and James Glass. DinoSR: Self-distillation and online clustering for self-supervised speech representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Andy T. Liu, Shang-Wen Yang, Po-Han Chi, Po-Chun Hsu, and Hung-yi Lee. Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders. In *ICASSP*, 2020b.

Andy T. Liu et al. Tera: Self-supervised learning of transformer encoder representation for speech. *IEEE/ACM TASLP*, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *ICLR*, 2019.

James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An asr corpus based on public domain audio books. In *ICASSP*, 2015.

Santiago Pascual, Mirco Ravanelli, Joan Serra, Antonio Bonafonte, and Yoshua Bengio. Learning problem-agnostic speech representations from multiple self-supervised tasks. In *INTERSPEECH*, 2019.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision, 2022. URL https://arxiv.org/abs/2212.04356.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. In *Interspeech*, 2019.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models. *arXiv preprint arXiv:2502.02013*, 2025.

Kenneth N. Stevens. *Acoustic Phonetics*. MIT Press, 1998.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. In *arXiv preprint arXiv:1807.03748*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Jeffrey S. Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57, 1985.

Shu wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y. Lin, Andy T. Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, Tzu-Hsien Huang, Wei-Cheng Tseng, Ko tik Lee, Da-Rong Liu, Zili Huang, Shuyan Dong, Shang-Wen Li, Shinji Watanabe, Abdelrahman Mohamed, and Hung yi Lee. Superb: Speech processing universal performance benchmark, 2021. URL https://arxiv.org/abs/2105.01051.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations*, 2022.

A Architecture Details

The architecture follows HuBERT (Hsu et al., 2021) for the encoder, with a single change: 6 Transformer (Vaswani et al., 2017) encoder layers in place of HuBERT-Base's 12. The predictor and cluster head are described below; both are used only during training and discarded afterwards.

A.1 CNN Feature Extractor

A 7-layer 1D convolutional feature extractor maps the raw 16 kHz waveform to a sequence of frame features. Kernel sizes and strides match HuBERT exactly: $(10, 5)$ for the first layer, $(3, 2)$ for layers 2–5, and $(2, 2)$ for layers 6–7. Each conv is followed by GELU; the first layer additionally uses fp32 group normalization. The total stride is 320 samples, yielding 20 ms frames at 16 kHz. Output channel width is 512. Following HuBERT, gradients flowing back through the CNN feature extractor are scaled by $\gamma_{\text{feat}} = 0.1$ to prevent the frontend from learning too quickly.

A.2 Transformer Stack

Six Transformer encoder layers, configured following HuBERT-Base except for layer count: 768-dimensional hidden state, 12 attention heads (head dimension 64), feed-forward dimension 3072, GELU activations, post-norm layout. Layer normalization is applied in fp32. A convolutional positional encoding (kernel 128, 16 groups, weight-normalized) is added to the encoder input before the Transformer stack runs. LayerDrop is applied at probability 0.05 per layer during training. Attention dropout, residual dropout, and activation dropout are each set to 0.1.

A.3 Encoder Mask Handling

The encoder accepts a binary frame-level mask alongside the waveform. After the CNN frontend and the post-extract projection, masked frames are zeroed out before the convolutional positional encoding and the Transformer stack run. The encoder thus produces representations of the visible context with no signal at masked positions; the mask token is injected later, by the predictor.

A.4 Predictor and Cluster Head

The predictor injects a learned 768-dimensional mask token at masked positions, adds learned positional embeddings (capacity for 750 frames, sufficient for 15 s of audio at 20 ms frames), and processes the result with a single Transformer encoder layer (8 heads, feed-forward dimension $2 \times 768 = 1536$, dropout 0.1, batch-first). A linear projection produces frame-level features that are passed to the cluster head.

The cluster head is a 3-layer MLP applied to per-frame features:

$$\text{Linear}(768 \to 768) \to \text{GELU} \to \text{Linear}(768 \to 768) \to \text{GELU} \to \text{Linear}(768 \to K),$$

where $K = 100$ in Phase 1 and $K = 500$ in Phase 2. The cluster head is shared between the visible-position path (encoder output, supervised by the KL loss in Phase 1 and early Phase 2) and the masked-position path (predictor output, always supervised by the KL loss in Eq. 2).

A.5 EMA Encoder

An architecturally identical copy of the encoder, $\bar{f}$, is initialized from the online encoder at the Phase 1$\to$2 boundary and maintained throughout Phase 2, with parameters updated by exponential moving average of the online encoder, in line with self-distillation practice (Grill et al., 2020; Baevski et al., 2022). In Phase 1 it is unused; in Phase 2 its outputs at the active layer $\ell^\star$ provide the input features on which the online GMM is fit, and its decay rate is switched periodically between two values as in Eq. 3.

Algorithm 1 S-JEPA training. Phase 1 uses a fixed MFCC GMM; Phase 2 uses the online encoder-feature GMM with adaptive layer selection and switched EMA decay.

Init: f_ϕ, h_ψ (with mask token + pos. embeddings), g_ω, GMM, active layer $\ell^\star$, EMA decay $\alpha_t \leftarrow \alpha_{\text{fast}}$
for each minibatch $\{\mathbf{x}\}$ **do**
  at the Phase 1→2 boundary: $\bar{\phi} \leftarrow \phi$; initialize GMM ($K$=500; k-means + EM) on $\bar{f}$ features at $\ell^\star$; re-initialize g_ω with K=500 outputs
  $\mathbf{x}_{\text{aug}}, \mathbf{x}_{\text{clean}} \leftarrow \text{Augment}(\mathbf{x})$
  $\mathbf{m} \leftarrow \text{MFCC}(\mathbf{x}_{\text{clean}})$ (Phase 1) **or** $\bar{f}(\mathbf{x}_{\text{clean}})$ at $\ell^\star$ (Phase 2) {no_grad}
  $\mathbf{q} \leftarrow \text{GMM.posterior}(\mathbf{m})$
  if Phase 2 **then**
    GMM ← EMA-update from $\mathbf{m}$, $\mathbf{q}$ {online; no gradient}
    if layer-check step **then**
      Compute $\text{erank}(Z^{(\ell)})$ for all ℓ; EMA-update per-layer scores; $\ell^\star \leftarrow \arg\max_\ell \text{erank}_\ell^{\text{ema}}$
    end if
  end if
  $mask \leftarrow \text{SampleBlockMask}()$; $z \leftarrow f_\phi(\mathbf{x}_{\text{aug}}, mask)$; $\hat{z} \leftarrow h_\psi(z, mask)$
  $\mathbf{p}_t \leftarrow \text{softmax}(g_\omega(\hat{z}_t))$ for $t \in \mathcal{M}$; $\mathbf{p}_t \leftarrow \text{softmax}(g_\omega(z_t))$ for $t \in \mathcal{V}$
  $S \leftarrow \mathcal{M} \cup \mathcal{V}$ (Phase 1, early Phase 2) **or** $\mathcal{M}$ (later Phase 2)
  $\mathcal{L} \leftarrow \frac{1}{|S|} \sum_{t \in S} \text{KL}(\mathbf{q}_t \| \mathbf{p}_t)$; update ϕ, ψ, ω via $\nabla\mathcal{L}$
  if Phase 2 **then**
    Flip $\alpha_t \in \{\alpha_{\text{fast}}, \alpha_{\text{slow}}\}$ per switching schedule (Eq. 3); $\bar{\phi} \leftarrow \alpha_t \bar{\phi} + (1 - \alpha_t)\phi$
  end if
end for
Return: f_ϕ {predictor, cluster head, GMM discarded}

A.6 Masking

Block-wise temporal masking samples contiguous spans until a target fraction of positions is masked. The encoder zeros out masked positions internally; the learned mask token is applied later, by the predictor, before its Transformer layers run.

B Training Algorithm

Algorithm 1 summarizes the full procedure.

C Phase 2 Transition: Masked-Only Loss and Augmentation Off

We started Phase 2 with the same loss configuration as Phase 1: KL applied at both masked and visible positions, with denoising augmentation enabled. Partway through Phase 2 we made two changes simultaneously, motivated by tracking downstream WER on a held-out development set:

1. **Dropping the visible-position loss.** Visible-position cluster predictions are an easier prediction problem (the encoder sees the input directly there) and may supply gradients that are largely redundant with what the masked-position loss already provides. Dropping the visible-position term ($\mathcal{S} = \mathcal{M}$) is consistent with the masked-only configuration used by data2vec (Baevski et al., 2022), wav2vec 2.0 (Baevski et al., 2020b), and HuBERT (Hsu et al., 2021).
2. **Disabling augmentation.** We set $p_{\text{noise}} = p_{\text{mix}} = 0$, so the encoder sees the same clean waveform that the EMA encoder feeds to the GMM. This is in tension with WavLM's (Chen et al., 2022) finding that denoising augmentation is helpful, and likely reflects an interaction specific to our setup: if the GMM target is computed from clean audio while the encoder sees augmented audio, there is an extra invariance demand on the encoder that may make masked-position prediction harder than necessary at this stage of training.

After both changes, downstream WER continued to decrease and the auto-layer mechanism settled at $\ell^\star = 4$. We did not run a controlled ablation toggling each change independently; see Section 6.1.

Table 3: Training hyperparameters used in our experiments. AdamW (Loshchilov & Hutter, 2019) is the optimizer for both phases.

	Phase 1 (MFCC GMM)	**Phase 2 (online GMM)**
Encoder backbone (shared across phases)		
Hidden dimension	768	768
Transformer layers	6	6
Attention heads	12	12
FFN dimension	3072	3072
CNN feature-extractor channels	512	512
CNN total stride / frame rate	320 / 50 Hz	320 / 50 Hz
Conv positional encoding (kernel, groups)	(128, 16)	(128, 16)
Dropout / attention / activation	0.1 / 0.1 / 0.1	0.1 / 0.1 / 0.1
LayerDrop	0.05	0.05
Feature-extractor gradient multiplier	0.1	0.1
GMM		
K (number of components)	100	500
Feature space	39-dim MFCC+Δ+$\Delta\Delta$	768-dim EMA encoder features
Fitting	k-means init (5 it.) + EM (20 it.)	online EMA of suff. stats
Online param EMA decay	— (frozen)	0.999
Layer-check cadence (auto-layer)	— (n/a)	every $\sim$ 10,000 steps
Optimization		
Optimizer	AdamW	AdamW
β_1, β_2	$(0.9, 0.99)$	$(0.9, 0.99)$
Weight decay	10^{-3}	10^{-3}
Learning rate	1×10^{-4}	2.5×10^{-5}
Batch size (global)	192	192
Max input duration	15 s	15 s
Compute environment (identical for both phases)		
GPUs	8 $\times$ NVIDIA B200 (179 GiB each)	
CPU	2 $\times$ Intel Xeon Platinum 8570	
Host RAM	$\approx$2 TB	
Operating system	Ubuntu 24.04.4 LTS (kernel 6.8.0)	
PyTorch / CUDA / driver	2.9.1 / 12.8 / 580.173.02	
Python	3.12.3	
EMA encoder $\bar{f}$ (periodically switched decay, Phase 2 only)		
α_{slow}	— (n/a)	0.9999
α_{fast}	— (n/a)	0.999
Switching cadence	— (n/a)	every $\sim$ 20,000 steps (avg.)
Masking		
Mask ratio	0.65	0.65
Mask span length (frames)	10	10
Loss configuration		
KL applied at	masked + visible	masked + visible (early), masked only (after)
Augmentation		
p_{noise}	0.25	0.25 (early), 0.0 (after)
p_{mix}	0.25	0.25 (early), 0.0 (after)
SNR range (noise mix), dB	$(-5, 20)$	$(-5, 20)$ when active
Energy ratio range (utterance mix), dB	$(-5, 5)$	$(-5, 5)$ when active

D Training Hyperparameters

Table 3 reports the full set of training hyperparameters. Values are taken from the launch commands actually used to produce the checkpoints reported in this paper.

E GMM DETAILS

E.1 POSTERIOR COMPUTATION

For the diagonal-covariance GMM the posterior is computed in closed form (Bishop, 2006):

$$q_k(\mathbf{m}) = \mathrm{softmax}_k\Big(\log \pi_k - \tfrac{D}{2}\log(2\pi) \\ - \tfrac{1}{2}\sum_{d=1}^{D}\Big[\log \sigma_{k,d}^2 + \tfrac{(m_d-\mu_{k,d})^2}{\sigma_{k,d}^2}\Big]\Big) \tag{4}$$

Variances are clamped at a small floor (10^{-6}) for numerical stability throughout.

E.2 PHASE 1 GMM FITTING

The Phase 1 GMM is fit once on 39-dimensional MFCC features (13 cepstral coefficients with first and second deltas, 20 ms frames) and held fixed thereafter. Fitting proceeds in three steps.

Feature collection via reservoir sampling. We stream audio from the training corpus and extract MFCC+delta features frame by frame. To bound memory, we maintain a frame reservoir: incoming frames are added until the reservoir reaches a target size, after which each new frame replaces a uniformly random reservoir entry with the appropriate probability (Vitter, 1985).

Mini-batch k-means initialization. The GMM means are initialized by running mini-batch k-means (MacQueen, 1967) over the reservoir for 5 iterations. Cluster variances are initialized as the per-cluster diagonal sample variances of their k-means members, and mixture weights as the empirical cluster proportions. Empty clusters are re-seeded from random reservoir frames.

EM refinement. Starting from the k-means initialization, we run 20 iterations of expectation-maximization (Dempster et al., 1977) for the full diagonal-covariance GMM. Sufficient statistics are accumulated in chunks over the reservoir to avoid materializing the full $N \times K$ responsibility matrix; M-step updates are applied at the end of each iteration.

E.3 ONLINE GMM UPDATES (PHASE 2)

In Phase 2, the GMM parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2\}$ are updated online from minibatch sufficient statistics. Given N feature vectors $X \in \mathbb{R}^{N\times D}$ from the EMA encoder's active layer, with their soft posteriors $Q \in \mathbb{R}^{N\times K}$, we compute responsibility-weighted means $\hat{\boldsymbol{\mu}}_k = \frac{1}{\hat{N}_k}\sum_n q_{n,k}\mathbf{x}_n$, variances $\hat{\boldsymbol{\sigma}}_k^2 = \frac{1}{\hat{N}_k}\sum_n q_{n,k}(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k)^2$, and weights $\hat{\pi}_k = \hat{N}_k / \sum_j \hat{N}_j$ where $\hat{N}_k = \sum_n q_{n,k}$. We then EMA-update:

$$\theta \leftarrow \alpha\theta + (1-\alpha)\hat{\theta}, \quad \theta \in \{\pi, \boldsymbol{\mu}, \boldsymbol{\sigma}^2\} \tag{5}$$

No gradient flows through the GMM at any point.

E.4 CHUNKED SOFT ASSIGNMENT

For large K and large batch sizes, soft posteriors are computed in chunks over K to bound peak memory (Algorithm 2).

Algorithm 2 Chunked Soft Assignment

Input: Features $X \in \mathbb{R}^{N \times D}$, GMM parameters $\{\pi_k, \mu_k, \sigma_k^2\}$, chunk sizes
Output: Posteriors $Q \in \mathbb{R}^{N \times K}$
for $n = 0$ to N step batch_size **do**
  for $k = 0$ to K step chunk_k **do**
    $\Delta \leftarrow X[n : n'] - \mu[k : k']$
    mahal $\leftarrow \sum_d \Delta_d^2 / \sigma_{k,d}^2$
    $\log p_{n,k} \leftarrow -\frac{D}{2} \log(2\pi) - \frac{1}{2} \sum_d \log \sigma_{k,d}^2 - \frac{1}{2}$mahal
  end for
  $Q[n : n'] \leftarrow \text{softmax}(\log \pi + \log p)$
end for

F Periodically Switched EMA Decay

This appendix gives the full details of the periodically switched EMA decay on the EMA encoder $\bar{f}$ used in Phase 2 (see the "Periodically switched EMA decay" paragraph of Section 3.3 for the conceptual description).

Schedule. The decay rate α_t is a piecewise-constant function of the step index taking one of two values. We hold the rate fixed at one value for some number of steps, then flip it to the other, then back. Intervals are not exactly uniform across training, but they average $T_{\text{switch}} \approx 20{,}000$ steps per interval. The target-encoder update is $\bar{\phi} \leftarrow \alpha_t \bar{\phi} + (1 - \alpha_t)\phi$, applied once per minibatch. We use $\alpha_{\text{fast}} = 0.999$ and $\alpha_{\text{slow}} = 0.9999$. There is no smoothing or interpolation between values.

Effective EMA timescale. The characteristic timescale at decay α is $\tau(\alpha) = 1/(1 - \alpha)$ steps. With our values, $\tau(\alpha_{\text{slow}}) = 10{,}000$ and $\tau(\alpha_{\text{fast}}) = 1{,}000$. The mean switching cadence ($\sim 20{,}000$ steps) is comfortably larger than either timescale, so within a single fast-rate interval the EMA encoder has time to substantially track the online encoder, and within a single slow-rate interval the EMA encoder has time to settle and provide approximately stationary targets.

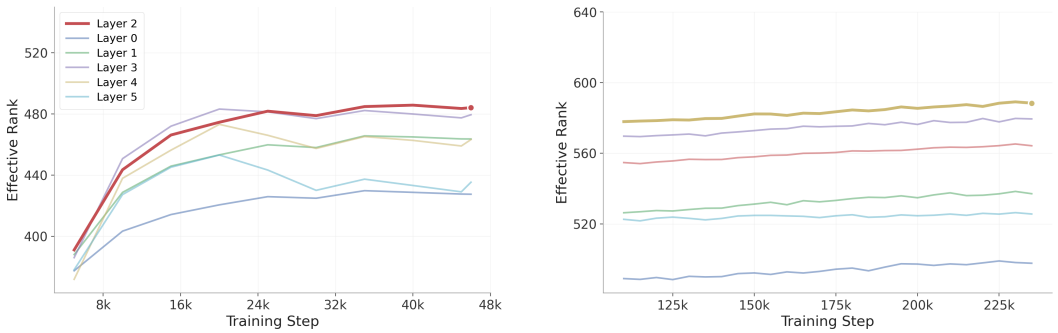
Interaction with online GMM updates. The online GMM is EMA-updated at every step from the EMA encoder's outputs at the active layer. When the schedule is at α_{fast}, the EMA encoder shifts faster, so the responsibility-weighted statistics that drive the GMM update reflect a more recent encoder, and the GMM's components track recent encoder improvements. When the schedule is at α_{slow}, the EMA encoder is approximately stationary, and so is the distribution from which the GMM's sufficient statistics are sampled.

Initialization and phase boundary. At the start of Phase 2, $\bar{f}$ is initialized from the Phase 1 encoder ($\bar{\phi} \leftarrow \phi$). Phase 2 begins under α_{fast}: at the Phase 1→2 boundary the online encoder has just been trained against MFCC GMM targets, and the online GMM (just initialized over EMA encoder features) benefits from being able to track the online encoder closely as it begins adapting to the new $K = 500$ targets.

G Effective Rank and Adaptive Layer Selection

Computation. For each transformer layer ℓ, we sample $N = 2{,}000$ frame embeddings from the EMA encoder's output at layer ℓ, center them, compute the singular values via SVD, normalize them to a probability distribution, and apply the effective-rank definition of Section 3.3. The per-layer values are EMA-smoothed across measurement intervals.

Cadence. Layer checks are performed at a fixed step cadence during Phase 2, set to roughly the natural EMA decay timescale (i.e., the number of steps over which the EMA-smoothed score is dominated by recent measurements). When the argmax changes, the GMM input layer switches at the start of the next minibatch with no other training discontinuity.



(a) **Phase 1.** Layer 2 settles as the most expressive layer, motivating its use as the initial GMM input layer for Phase 2.

(b) **Phase 2.** Under the online GMM, layer 4 dominates from the first measured checkpoint and the L4–L2 gap widens.

Figure 4: Per-layer effective rank across both phases of training. The horizontal axis shows training step.

Calibration study. For the calibration reported in Section 3.3, we trained a small CTC-style ASR head (Graves et al., 2006) on top of each of the 6 transformer layers in turn (encoder frozen, head trained from scratch on a small held-out labeled subset for 1,000 steps), and recorded each head's word error rate on a held-out development set. The lowest-WER layer coincided with the layer of highest effective rank computed on a fixed batch of features. We use this coincidence to justify effective rank — computable in seconds with no downstream training — as the layer-selection signal during Phase 2. The check is a single snapshot taken at the Phase 1→2 boundary; we did not repeat it during Phase 2.

H Probing Protocol

This appendix describes the probing protocol used in Section 4.2. All probes are trained on frozen encoder features extracted from each method (S-JEPA, HuBERT-Base, WavLM-Base), with no fine-tuning of the encoder.

Features. For each utterance, 768 dimensional frame-level features are extracted from the final encoder layer at 20 ms resolution. For utterance-level tasks (Speaker ID, Gender, Chapter ID), frame features are mean-pooled over the utterance. For Phoneme Classification, frame features are used directly without pooling, with each frame assigned the phoneme label of the interval containing its center timestamp, using forced alignments from the `gilkeyio/librispeech-alignments` dataset.

Linear probe. We use L_2-regularized multinomial logistic regression (scikit-learn's `LogisticRegression`, $C = 1.0$, lbfgs solver, up to 2000 iterations). Features are standardized to zero mean and unit variance using training-split statistics. We use the same probe configuration across all three encoders (S-JEPA, HuBERT-Base, WavLM-Base).

MLP probe. We used an MLP with 2 hidden layers each with 256 dimensions, ReLU activations, and 0.1 dropout after each hidden layer. We used ADAM, lr = 1e-3, and CrossEntropyLoss with a batchsize of 512, for 20 epochs.

Splits. All probes are trained and evaluated on LibriSpeech `dev-clean` using random 80/20 train–test splits. Utterance-level tasks (Speaker ID, Gender, Chapter ID) use ten splits; frame-level Phoneme Classification uses three. Reported numbers in Table 2 are mean ± standard deviation across splits.

Tasks. Speaker ID is multiclass classification over the speakers in LibriSpeech `dev-clean`. Gender Classification is binary (male/female). Chapter ID is multiclass classification over book

chapters; since each chapter is read by a single speaker but speakers read multiple chapters, distinguishing chapters by the same speaker requires recording-session and prosodic cues beyond speaker identity. Phoneme Classification is frame-level multiclass classification over the 67 phonemes in the forced alignments.

I Augmentation Details

The denoising augmentation scheme follows WavLM (Chen et al., 2022). The encoder sees the augmented waveform; the GMM-target side computes its features from the clean waveform.

I.1 Energy-Based SNR Mixing

Given clean signal x and noise n, mixing at target SNR:

$$E_x = \frac{1}{|x|}\sum_i x_i^2, \quad E_n = \frac{1}{|n|}\sum_i n_i^2, \tag{6}$$

$$\alpha = \sqrt{\frac{E_x}{10^{\text{SNR}/10} \cdot E_n}}, \tag{7}$$

$$x_{\text{aug}} = x + \alpha \cdot n \tag{8}$$

I.2 Utterance Mixing

A segment from another utterance is mixed with energy-based weighting:

$$\mathbf{x}_{\text{mix}}[t_1 : t_2] = \mathbf{x}_1[t_1 : t_2] + \beta \cdot \mathbf{x}_2[s_1 : s_2], \tag{9}$$

where $\beta = \sqrt{E_1 \cdot 10^{\rho/10}/E_2}$ with energy ratio ρ sampled from a fixed range. Concretely:

1. Sample mix length L_{mix} as a random fraction of the primary length.
2. Sample start positions t_1, t_2 in the primary and secondary signals.
3. Extract regions $r_1 \leftarrow x_1[t_1 : t_1 + L]$, $r_2 \leftarrow x_2[t_2 : t_2 + L]$ and compute energies E_1, E_2.
4. Sample energy ratio $\rho \sim$ Uniform over a fixed dB range.
5. Compute β and mix $x_1[t_1 : t_1 + L] \leftarrow r_1 + \beta \cdot r_2$.

J SUPERB Evaluation Protocol

We follow the standard SUPERB (wen Yang et al., 2021) protocol: the pre-trained encoder is frozen and small task-specific heads are trained on top. Across tasks, the encoder receives raw waveform input and the task-specific head consumes its frame-level outputs (or a learned weighted combination across encoder layers, depending on the task). Reported numbers are in Table 1, with the full comparison in Table 4.

Table 4: Frozen-encoder evaluation on the SUPERB benchmark: ASR (WER on LibriSpeech `test-clean`), emotion recognition (ER, accuracy), and slot filling (SF, F1 and CER). All entries use the standard SUPERB protocol with greedy CTC decoding for ASR. Baseline numbers are from the public SUPERB leaderboard. Parameter counts are inference-time encoder only. Best result within the sub-90M block in bold. Full version of Table 1.

Method	**# Params**	**ASR** WER (↓)	**ER** Acc. (↑)	**SF** F1 (↑)	**SF** CER (↓)
Sub-90M baselines					
modified CPC (van den Oord et al., 2018)	1.8M	20.18	60.96	71.19	49.91
APC (Chung et al., 2019)	4.1M	21.28	59.33	70.46	50.89
VQ-APC (Chung et al., 2020)	4.6M	21.20	59.66	68.53	52.91
PASE+ (Pascual et al., 2019)	7.8M	25.11	57.86	62.14	60.17
NPC (Liu et al., 2020a)	19.4M	20.20	59.08	72.79	48.44
TERA (Liu et al., 2021)	21.3M	18.17	56.27	67.50	54.17
DistilHuBERT (Chang et al., 2022)	23.5M	13.37	63.02	82.57	35.59
wav2vec (Schneider et al., 2019)	32.5M	15.86	59.79	76.37	43.71
vq-wav2vec (Baevski et al., 2020a)	34.2M	17.71	58.24	77.68	41.54
Mockingjay (Liu et al., 2020b)	85.1M	22.82	50.28	61.59	58.89
DeCoAR 2.0 (Ling & Liu, 2020)	89.8M	13.02	62.47	**83.28**	34.73
S-JEPA (ours)	**51.8M**	**12.10**	**64.83**	83.05	**33.17**
Base-class (≈95M)					
HuBERT Base (Hsu et al., 2021)	94.7M	6.42	64.92	88.53	25.20
WavLM Base (Chen et al., 2022)	94.7M	6.21	65.94	89.38	22.86
WavLM Base+ (Chen et al., 2022)	94.7M	5.59	68.65	90.58	21.20
wav2vec 2.0 Base (Baevski et al., 2020b)	95.0M	6.43	63.43	88.30	24.77
DinoSR (Liu et al., 2023)	94.7M	4.71	—	88.83	23.57