STREAMDC: CONDENSING TIME SERIES INTO A SHARED TEMPORAL MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-series dataset condensation compresses a training set into a few synthetic sequences. Under a tight signal storage budget, these sequences must preserve diverse temporal patterns and their useful positional variation. We introduce StreamDC, which learns a class-segmented temporal memory and reuses its signal fragments through overlapping training windows. Linear windows slide across the memory with targets given by temporal class occupancy; class-local cyclic windows wrap from each segment's end to its beginning, adding within-class temporal offsets. All windows jointly optimize the same stored values through an existing teacher-guided condensation objective. After synthesis, denser decoding supplies additional teacher-labeled windows without increasing signal storage. This construction makes temporal reuse part of synthesis and applies across condensation backends. Across five time-series classification datasets, three backends, and two storage budgets, StreamDC achieves higher mean test accuracy than independent-sequence synthesis at equal signal storage in all 30 configurations, with an average gain of 19.5 percentage points. With ShapeCond, StreamDC also averages 12.7 percentage points higher test accuracy than TexTSC across ten dataset–budget settings.

1 INTRODUCTION

Dataset condensation learns a small synthetic training set that preserves the utility of a much larger dataset (Wang et al., 2018; Sachdeva & McAuley, 2023). For time series, the limited storage must capture both discriminative patterns and their useful temporal variation. A characteristic event may occur near the beginning or end of an observation window without changing its class. Storing a separate synthetic sequence for each placement spends scarce coordinates on repeated content. This motivates our central question: *how can a tiny synthetic set present the same class evidence at different temporal positions without storing a separate copy for each placement?*

Existing time-series condensation methods preserve temporal and spectral information or local motifs through their objectives (Liu et al., 2024; Miao et al., 2024; Peng et al., 2026; Hosseinzadeh et al., 2026). Multiple-view supervision is also established: differentiable Siamese augmentation coordinates real and synthetic transformations, and CondTSC supervises frequency-perturbed synthetic views (Zhao & Bilen, 2021; Liu et al., 2024). We study a complementary choice: how stored temporal fragments are shared across training windows. A sliding window alone has a limitation at the smallest signal storage budget. When each class stores exactly one window's worth of signal, there is only one window fully inside that class segment. Any other valid linear window crosses a class boundary; taking smaller steps cannot create another pure-class window.

StreamDC addresses this limitation by learning a shared temporal memory with two kinds of overlapping windows (Figure 1). We concatenate class-specific segments under the same signal storage budget as an independent synthetic set. Linear windows slide along this memory: windows inside a segment keep its class label, while those crossing a boundary receive targets proportional to each class's duration. Class-local cyclic windows additionally wrap from a segment's end to its beginning under the same class target. For a one-window segment, this is a circular shift; for a longer segment, it joins its endpoint portions into a training window. These reads expose different positions and neighboring content using the same stored values.

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.

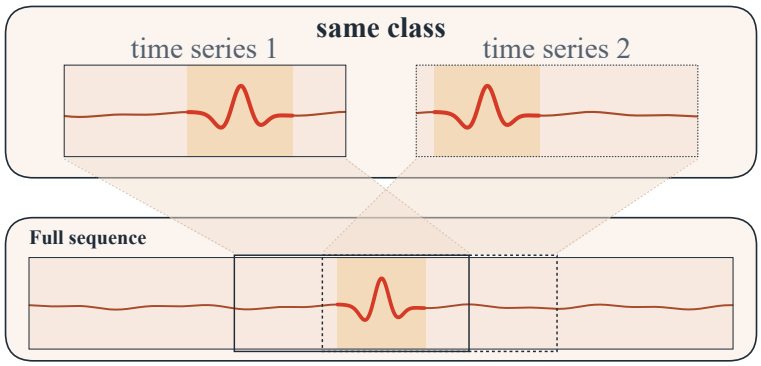

Figure 1: **Temporal reuse motivates a shared memory.** Two overlapping windows of one longer signal place the same local pattern at different positions while retaining the same class. Storing their shared signal once avoids duplicating the overlapping content. Waveforms and highlighted offsets are illustrative.

Both window families participate in synthesis. If a fragment appears in several windows, their gradients update the same memory coordinates, allowing that fragment to adapt to its multiple uses. After synthesis, we freeze the memory, read it at a finer stride, and train a fresh student using teacher-generated targets. StreamDC retains each condensation backend's loss components and student-supervision protocol, so temporal reuse can be studied across different objectives.

To our knowledge, we are the first to explicitly formulate and study the *storage–placement tension* in time-series dataset condensation: under a tight signal storage budget, the same discriminative temporal content may need to appear at multiple window-relative positions without repeated storage. Our contributions are threefold. First, we formulate the storage–placement tension in time-series dataset condensation. Second, we introduce StreamDC, a shared temporal memory jointly optimized through global linear and class-local cyclic readouts, with separate control over synthesis and decoding densities. Third, we evaluate its practical value through controlled comparisons and experiments across five datasets, three teacher-guided backends, and two storage budgets. StreamDC achieves higher mean accuracy than independent-sequence synthesis in all 30 main configurations, with gains of $+1.7$ to $+44.8$ percentage points.

2 Related Work

Time-series dataset condensation. CondTSC and TimeDC preserve temporal and spectral structure through dual-domain matching and decomposition with trajectory matching, respectively (Liu et al., 2024; Miao et al., 2024). ShapeCond emphasizes discriminative shapelets, TexTSC matches class-specific teacher-feature statistics in time and frequency, and CondTSF adapts condensation to forecasting (Peng et al., 2026; Hosseinzadeh et al., 2026; Ding et al., 2024). Contextual views also support time-series representation learning at timestamp and subsequence levels, as in TS2Vec (Yue et al., 2022). Multiple-view supervision is established in condensation: differentiable Siamese augmentation coordinates real and synthetic transformations, while CondTSC supervises Fourier-perturbed views (Zhao & Bilen, 2021; Liu et al., 2024). StreamDC specifies overlapping temporal readouts whose occurrences jointly update the same stored coordinates. Our focus is which temporal offsets and class contexts a fixed memory can expose.

Condensation objectives and teacher guidance. Dataset distillation learns compact training data (Wang et al., 2018; Sachdeva & McAuley, 2023) through gradient matching (Zhao et al., 2021), trajectory matching (Cazenavette et al., 2022; Guo et al., 2024), distribution matching (Zhao & Bilen, 2023), or kernel-based meta-learning (Nguyen et al., 2021). SRe^{2}L separates teacher training, recovery, and relabeling (Yin et al., 2023); G-VBSM diversifies recovery with multiple network backbones and statistical descriptors (Shao et al., 2024); RDED composes selected real patches and obtains teacher labels (Sun et al., 2024). DELT targets within-class diversity through staged

optimization of synthetic subsets (Shen et al., 2025). StreamDC complements the choice of synthesis objective: we use ShapeCond, SRe^{2}L, and G-VBSM as condensation backends, retaining their loss components while jointly optimizing shared temporal readouts.

Compact representations and multiple readouts. IDC derives examples through multi-formation (Kim et al., 2022), addressable memories combine shared learned bases through learned addressing functions (Deng & Russakovsky, 2022), and LADD adds local-view supervision to distilled images (Kang et al., 2025). Rate-utility distillation jointly optimizes latent representations and their storage cost (Bao et al., 2025). PoDD jointly optimizes overlapping image patches from a shared poster, with targets obtained from a fixed class layout or a learned spatial label field (Shul et al., 2025). StreamDC uses a class-segmented temporal memory with global linear reads and class-local cyclic reads. This combination exposes cross-class compositions and additional within-class temporal origins, with separate strides for synthesis and decoding. Appendix C.1 details the readout structure. Linear-window targets measure temporal class occupancy, paralleling CutMix's area-based labels (Yun et al., 2019); they define synthetic supervision without assuming that adjacent classes represent physical transitions.

3 StreamDC: Learning a Shared Temporal Memory

StreamDC learns one memory through overlapping windows, then decodes the frozen memory more densely for student training (Figure 2). We call the underlying teacher-guided condensation method a *backend*; its objective is retained throughout.

3.1 Memory and temporal readouts

Let $\mathcal{D}_{\text{train}}$ contain length-L time series with d channels and K classes. For a signal storage budget of $m \geq 1$ sample equivalents per class, with integer m, set $B = mL$ and define

$$S = [S_1|\cdots|S_K] \in \mathbb{R}^{d\times KB}, \qquad S_j \in \mathbb{R}^{d\times B}, \tag{1}$$

where segment j represents class π_j for a class permutation π. This stores dKB values, exactly as many as Km independent length-L examples at the same precision. Each segment is initialized from m uniformly sampled training examples of its class. Using Euclidean distances between the means of these sampled examples, we construct a greedy nearest-neighbor path from every starting class and retain the shortest. This fixes which classes are neighbors during synthesis.

Linear windows. Using zero-based indices and half-open slices, the global starts and readouts at synthesis stride s are

$$\mathcal{G}_s = \{ks : k \in \mathbb{Z}_{\geq 0},\ ks + L \leq KB\}, \qquad W_t^{\text{lin}}(S) = S[:, t : t + L]. \tag{2}$$

A window may lie within one class or cross into its neighbor. In both cases its synthesis target is the fraction of time occupied by each class. With $I_j = [(j-1)B, jB)$,

$$q_{t,c}^{\text{syn}} = \frac{1}{L}\sum_{j=1}^{K} \mathbf{1}\{\pi_j = c\}\,\big|[t, t+L) \cap I_j\big|. \tag{3}$$

Here $|\cdot|$ counts time steps. Since $B \geq L$, at most two classes contribute, and a within-segment window receives a one-hot target. Cross-class windows are synthetic compositions; occupancy specifies their supervision, not an observed transition probability.

Class-local cyclic windows. To add same-class windows across a segment's endpoints, we include the local starts

$$\mathcal{H}_s = \{ks : k \in \mathbb{Z}_{\geq 0},\ ks < B,\ ks + L > B\}, \tag{4}$$

Only starts with $u + L > B$ wrap across the segment endpoints. For $u \in \mathcal{H}_s$, the readout and one-hot target are

$$W_{j,u}^{\text{cyc}}(S)[:, r] = S_j[:, (u + r) \bmod B], \quad r = 0, \ldots, L-1, \qquad q_{j,u}^{\text{syn}} = e_{\pi_j}. \tag{5}$$

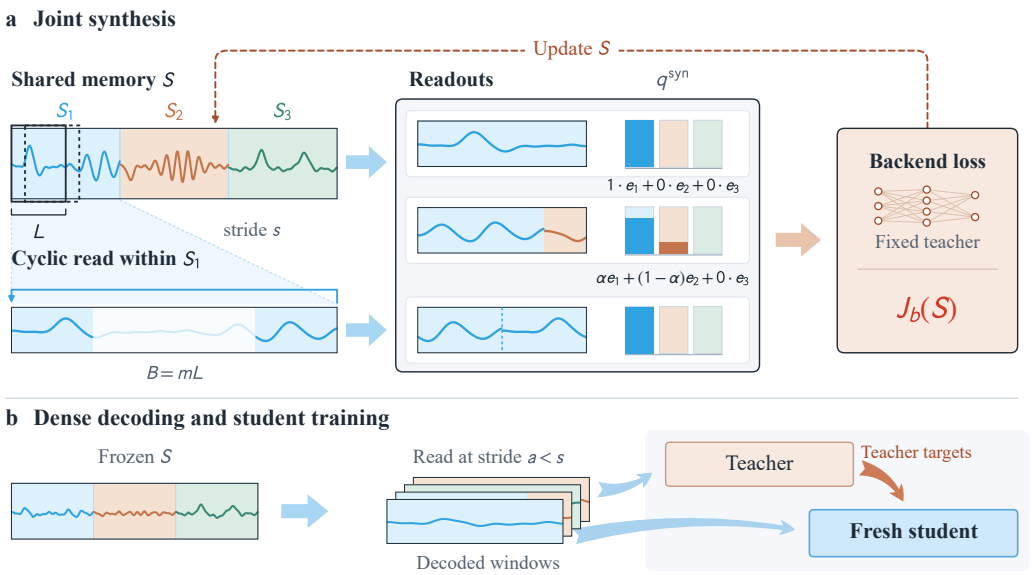

Figure 2: **StreamDC synthesis and decoding. (a)** Linear windows read the shared memory at stride s; class-local cyclic windows wrap around each full segment ($B = mL$). Highlighted endpoint portions form one length-L cyclic window; its faint interior is not read. Cross-class targets use temporal occupancy α; pure windows keep one class target. All reads jointly update shared values. **(b)** A finer stride a reads the frozen memory to supply windows and teacher targets for student training. Waveforms and offsets are schematic.

At $m = 1$, these are circular shifts of the stored sequence. At larger budgets, they join endpoint portions of the longer class memory. Together with pure linear windows, they give equal occurrence counts for all stored positions when $s \mid L$ (Lemma 1). This is a discrete constant-overlap-add property (Smith, 2011); it does not assume periodic real recordings.

The hybrid family $\mathcal{V}_s$ combines all global linear windows and these cyclic extras. Cycles are class-local: the last class is not connected to the first. We use $s = \lfloor L/2 \rfloor$ by default. The index sets also cover non-divisible lengths, with linear starts anchored globally and cyclic starts anchored within each segment; uniform coverage need not hold then.

3.2 Joint Synthesis

Let $X_s(S)$ stack the windows in $\mathcal{V}_s$ and Q_s their targets. Backend b learns the memory with

$$J_b(S) = \mathcal{L}_b\big(X_s(S), Q_s; \mathcal{T}_b, \xi\big), \tag{6}$$

where $\mathcal{T}_b$ is its pretrained teacher collection and ξ contains its statistics, state, and stochastic choices. The backend's loss components, including batch-coupled feature or normalization statistics, are evaluated on the hybrid windows. For the backends used here, the classification component for the teacher f_b selected at the current synthesis step is soft-target cross-entropy,

$$\mathcal{L}_{\text{cls}} = -\frac{1}{|\mathcal{V}_s|} \sum_{v \in \mathcal{V}_s} \sum_{c=1}^{K} q_{v,c}^{\text{syn}} \log \operatorname{softmax}\big(f_b(W_v(S))\big)_c. \tag{7}$$

Every occurrence of a fragment updates the same memory coordinates, coupling supervision across windows without adding learned parameters. We retain the snapshot with the lowest observed per-iteration synthesis objective. Appendix C.2 gives each backend's complete losses and optimization settings.

3.3 DENSE DECODING AND STUDENT TRAINING

After synthesis, we freeze S and apply the same readout rules at a finer stride a. This supplies additional offsets while changing the number and pure/cross-class proportions of windows. The dominant class $y_v^{\mathrm{dom}} = \min \arg\max_c q_{v,c}^{\mathrm{syn}}$ determines grouping and loader oversampling, with smallest-class-index tie breaking.

Teacher predictions provide targets for each decoded window. For the backends used here, the student objective has the form

$$\ell_b^{\mathrm{stu}}(x) = D_b\big(f_\theta(x),\, \mathcal{T}_b(x)\big) + \beta_b\ \mathrm{CE}\left(f_\theta(x), y^{\mathrm{dom}}\right), \tag{8}$$

where D_b is its distillation criterion (e.g., temperature-scaled KL divergence or logit regression), and β_b controls an optional dominant-class term. Thus occupancy targets supervise synthesis, while teacher predictions supervise downstream learning; Appendix C.2 specifies the protocols. Equal storage refers to the dKB signal values. Decoded-window counts and runtime costs are reported in Appendix B.

4 ANALYSIS

The analysis clarifies shared-coordinate optimization, cyclic coverage, and the distinct roles of synthesis and decoding. The coverage and risk results rely on assumptions not enforced throughout our experiments and provide no general guarantee of downstream performance. Experiments assess whether these mechanisms remain useful in broader practical settings. Proofs and full conditions appear in Appendix A.

4.1 TEMPORAL CONSISTENCY AND CYCLIC COVERAGE

Overlapping linear windows agree on their common fragment: for valid starts $t, t+\delta$ with $0 < \delta < L$,

$$W_t^{\mathrm{lin}}(S)[:, \delta : L] = W_{t+\delta}^{\mathrm{lin}}(S)[:, 0 : L-\delta].$$

Thus each fragment can occupy different window-relative positions while remaining the same learnable signal. For any fixed readout family $\mathcal{V}$, write the stacked windows as $\mathbf{x} = A_{\mathcal{V}}\,\mathrm{vec}(S)$, where each row of $A_{\mathcal{V}}$ selects one stored coordinate. For any differentiable joint loss $J(\mathbf{x})$,

$$\nabla_{\mathrm{vec}(S)} J = A_{\mathcal{V}}^\top \nabla_{\mathbf{x}} J, \qquad A_{\mathcal{V}}^\top A_{\mathcal{V}} = \mathrm{diag}(\boldsymbol{\kappa}), \tag{9}$$

where κ_i counts occurrences of coordinate i. Gradients from all contexts, including batch-coupled loss terms, therefore accumulate on shared coordinates. Additional readouts introduce no stored parameters: $\mathrm{rank}(A_{\mathcal{V}}) \le dKB$.

Lemma 1 (Aligned cyclic completion)**.** *Suppose $s \mid L$ and $B = mL$, and write $q = L/s$. In each class segment, the global linear grid contains $mq - q + 1$ pure windows. Cyclic extras add the $q-1$ missing starts of the local cyclic grid, yielding mq pure windows and exactly q occurrences of every stored time position. The largest gap between pure starts around the class cycle changes from L to s.*

For the pure family, Equation (9) becomes $A_{\mathrm{p}}^\top A_{\mathrm{p}} = qI$: all stored positions have equal occurrence counts. This does not imply equal gradient magnitudes or seam smoothness. At $s = L/2$, the pure-window count rises from $2m-1$ to $2m$, so the relative increase is largest at small budgets. At $m = 1$, the extra view changes the origin of the single stored sequence; no linear stride can supply its end-to-start composition under a pure-class target. For the default floor-based strides, divisibility can fail: global and class-local grids may have different phases, and uniform coverage is not guaranteed (Appendix A.1).

4.2 DENSE DECODING AS TEMPORAL DISCRETIZATION

Fix the learned memory and per-window student loss, including teacher targets. Finer decoding changes both temporal resolution and the proportion of cross-class contexts. For $K \ge 2$, $a \mid L$, and $q = L/a \ge 2$, the uniform hybrid decoder contains

$$N_{\mathrm{p}} = Kmq, \qquad N_{\mathrm{c}} = (K-1)(q-1), \qquad \rho_a = \frac{N_{\mathrm{c}}}{N_{\mathrm{p}} + N_{\mathrm{c}}}, \tag{10}$$

where p and c denote pure and cross-class readouts. Its risk is $R_a = (1-\rho_a)R_{\mathrm{p},a} + \rho_a R_{\mathrm{c},a}$; let R_* be the stride-one reference with weight ρ_*. If losses lie in $[0, M]$ and branchwise risk discrepancies are bounded by $\epsilon_{\mathrm{p}}, \epsilon_{\mathrm{c}}$, respectively, then Proposition 1 gives

$$|R_a(\theta) - R_*(\theta)| \le (1-\rho_a)\epsilon_{\mathrm{p}} + \rho_a\epsilon_{\mathrm{c}} + M|\rho_a - \rho_*|. \tag{11}$$

The first two terms measure offset discretization; the last measures the changed mixture. Under losses Lipschitz in normalized cyclic and boundary offsets, the respective branch-error bounds scale as $(a-1)/B$ and $(a-1)/L$ (Appendix A.2). Class balancing adds a sampling-measure discrepancy, and a bound uniform over students transfers to approximate minimizers (Appendix A.3). These statements concern the synthetic memory's readout risk; they do not guarantee monotonic gains from denser decoding or improved real-data accuracy.

4.3 Joint synthesis shapes unseen temporal contexts

Post-hoc decoding fixes the memory; joint synthesis lets its coordinates respond to the additional contexts. Let J_0 be the complete linear-only synthesis objective and $C = J_{\mathrm{hybrid}} - J_0$, including changes in normalization and batch-coupled terms. Consider $J_\lambda = J_0 + \lambda C$. At a stationary point S_0 with $H = \nabla^2 J_0(S_0) \succ 0$, local smoothness gives a continuing minimizer S_λ. For a smooth held-out-readout risk R_{d} evaluated with a fixed model, Proposition 2 gives

$$R_{\mathrm{d}}(S_\lambda) - R_{\mathrm{d}}(S_0) = -\lambda \left\langle \nabla R_{\mathrm{d}}(S_0), H^{-1}\nabla C(S_0) \right\rangle + O(\lambda^2). \tag{12}$$

A positive inner product yields a decrease for sufficiently small positive λ. This conditional, fixed-model result describes how synthesis can improve unseen offsets; it does not establish the effect of the full hybrid objective or subsequent student training. Conversely, at $m = 1$, an exactly cyclically invariant additive loss has the same objective and gradient on cyclic views as on equally weighted duplicate-pure views. This degeneracy need not hold for batch-coupled terms (Appendix A.5).

5 Experiments

5.1 Setup

Datasets and budgets. We evaluate on HAR (Anguita et al., 2013), FacesUCR and Electric (Dau et al., 2019), Sleep (Goldberger et al., 2000), and Insect (Chen et al., 2014) at signal storage budgets of $m \in \{1, 5\}$ samples per class (SPC), with each sample equivalent to a length-L sequence. Appendix C.2 details dataset dimensions, prepared train/validation/test splits, and implementation settings.

Backends. We use time-series implementations of three teacher-guided condensation backends: SRe^{2}L-style (Yin et al., 2023), G-VBSM (Shao et al., 2024), and ShapeCond (Peng et al., 2026). StreamDC retains each backend's loss components, teacher states, and student-supervision protocol, while changing signal organization and readouts and using occupancy targets for cross-class synthesis windows. The corresponding baseline (*Indep.*) synthesizes m independent length-L sequences per class at equal signal storage.

Protocol. Default strides are $s = \lfloor L/2 \rfloor$ for synthesis and $a = \max(\lfloor L/16 \rfloor, 4)$ for decoding; the similarity-based class order is fixed before synthesis (Section 3). Students learn from teacher-labeled readouts and are selected by validation loss. Within-backend comparisons report held-out test accuracy as mean $\pm$ standard deviation over nine runs: three teacher initializations with three synthesis repetitions each.

5.2 Main results

At equal signal storage, StreamDC achieves higher mean accuracy in all 30 dataset–backend–budget configurations (Table 1), averaging 19.5 points, with gains from $+1.7$ points (Sleep, SPC 1, G-VBSM) to $+44.8$ points (Insect, SPC 5, SRe^{2}L-style). Across budgets, StreamDC at SPC 1 also exceeds the SPC 5 Indep. mean in 12 of 15 dataset–backend pairs, using one fifth of the signal storage. These comparisons concern reported means.

Table 1: Test accuracy (%) at equal signal storage. Bold marks the higher mean within each backend. *Full* is a supervised full-data reference over three seeds.

		SRe^{2}L-style		G-VBSM		ShapeCond		
Dataset	SPC	Indep.	StreamDC	Indep.	StreamDC	Indep.	StreamDC	Full
HAR	1	61.05±3.37	**80.73±1.62**	53.36±9.96	**80.85±1.61**	72.77±2.72	**86.35±2.10**	95.51±0.47
	5	78.85±2.09	**93.73±0.67**	76.59±2.17	**92.67±0.59**	86.41±2.10	**93.92±0.65**	
FacesUCR	1	73.57±4.47	**94.53±1.13**	68.54±2.98	**87.77±2.51**	72.60±1.82	**93.80±0.54**	97.74±0.21
	5	90.54±1.42	**96.09±0.59**	88.79±1.58	**93.75±0.72**	90.25±1.21	**95.40±0.62**	
Electric	1	45.04±9.08	**56.23±11.30**	38.14±5.83	**62.34±2.69**	48.76±9.65	**64.29±3.23**	81.65±1.71
	5	60.58±1.47	**75.86±2.19**	60.47±2.61	**72.25±1.05**	59.56±1.44	**74.63±2.76**	
Sleep	1	24.93±5.31	**49.39±12.55**	38.52±7.44	**40.21±12.25**	17.41±8.20	**47.86±13.14**	75.01±0.53
	5	35.62±10.15	**65.80±3.03**	38.69±6.43	**67.98±1.91**	41.43±6.37	**68.07±5.13**	
Insect	1	11.27±0.79	**28.42±9.14**	10.41±0.64	**24.30±2.05**	12.48±1.04	**29.96±6.25**	68.54±0.72
	5	11.72±1.29	**56.48±1.28**	10.02±0.59	**47.21±1.65**	15.02±1.55	**56.94±1.39**	

Table 2: Reproduced results of existing methods (test accuracy, %). StreamDC uses the ShapeCond backend. Methods share data splits and retain their training protocols (Appendix C.6); bold marks the largest reported mean in each row.

Dataset	SPC	Random	K-center	CondTSC	TexTSC	StreamDC
HAR	1	54.03±5.76	64.25±2.18	68.88±0.61	76.44±1.65	**86.35±2.10**
	5	72.49±0.70	73.89±3.61	86.01±4.27	85.15±4.93	**93.92±0.65**
FacesUCR	1	41.65±5.38	55.80±3.22	74.22±1.53	61.56±6.01	**93.80±0.54**
	5	80.16±0.84	80.53±2.06	87.92±0.98	79.79±1.02	**95.40±0.62**
Electric	1	37.59±6.72	33.69±3.67	43.39±6.21	55.57±1.60	**64.29±3.23**
	5	53.82±1.52	43.89±1.84	55.10±0.79	60.06±4.52	**74.63±2.76**
Sleep	1	23.75±13.02	14.25±5.35	37.63±3.03	33.91±3.34	**47.86±13.14**
	5	33.25±11.93	41.21±1.84	29.57±1.28	37.82±3.46	**68.07±5.13**
Insect	1	10.85±0.48	11.16±0.44	34.66±1.16	**51.58±0.90**	29.96±6.25
	5	11.00±0.31	19.34±1.10	47.25±1.70	42.68±12.36	**56.94±1.39**

5.3 Comparison with existing methods

Table 2 places StreamDC with the ShapeCond backend alongside Random, K-center, CondTSC (Liu et al., 2024), and TexTSC (Hosseinzadeh et al., 2026). All methods use the same data splits while retaining their own architectures and training protocols (Appendix C.6), complementing the within-backend comparisons above. StreamDC records the largest mean in nine of ten settings; TexTSC has the highest mean on Insect at SPC 1.

5.4 Window design and synthesis controls

Table 3 studies ShapeCond on HAR and Sleep at equal signal storage.

Window composition ablation. Removing cyclic windows leaves global linear reads; removing cross-class windows uses class-local linear reads with cyclic completion. Both variants apply at synthesis and decoding, retaining their default strides (local grids are specified in Appendix C.3). At SPC 1, removing either family lowers mean accuracy on HAR; on Sleep, omitting cross-class windows produces the larger reduction. At SPC 5, the variants have similar mean accuracies on both datasets.

Joint synthesis versus post-hoc concatenation. We concatenate independently synthesized sequences and apply the same StreamDC readout and teacher-labeling rules. Post-hoc decoding improves over Indep., while joint synthesis yields further mean gains of 3.37/0.51 points on HAR and

Table 3: ShapeCond controls (test accuracy, %). Reference rows repeat Table 1. Post-hoc decoding uses independently synthesized signals; replication matches StreamDC's decoded-window count before class balancing.

Dataset	Variant	SPC 1	SPC 5
HAR	Indep. (reference)	72.77±2.72	86.41±2.10
	StreamDC (full)	**86.35±2.10**	**93.92±0.65**
	w/o cyclic windows	76.73±4.16	93.79±0.64
	w/o cross-class windows	82.51±3.75	93.95±0.51
	post-hoc concat. + StreamDC decode	82.98±2.81	93.41±0.53
	indep., replicated (pre-balance count)	71.14±2.58	85.68±2.83
Sleep	Indep. (reference)	17.41±8.20	41.43±6.37
	StreamDC (full)	**47.86±13.14**	**68.07±5.13**
	w/o cyclic windows	46.84±13.38	67.16±5.55
	w/o cross-class windows	35.78±8.24	68.76±3.56
	post-hoc concat. + StreamDC decode	38.30±8.23	66.29±4.44
	indep., replicated (pre-balance count)	17.36±6.66	41.11±7.64

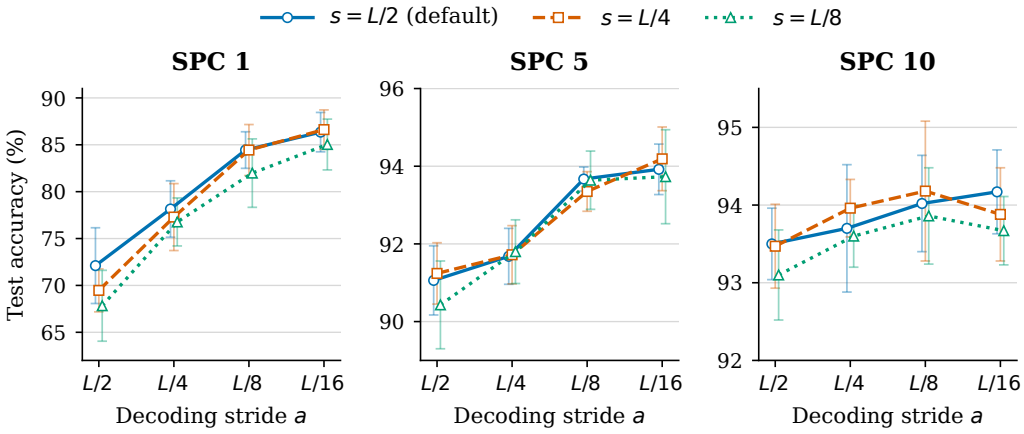

Figure 3: HAR–ShapeCond stride sweep. Points and error bars show mean test accuracy and one standard deviation. Decoding becomes denser left to right; vertical-axis ranges differ by SPC budget.

9.56/1.78 on Sleep at SPC 1/5. These comparisons assess the practical value of joint synthesis in the full training pipeline, beyond the local, fixed-model setting of Proposition 2.

Replication of independent samples. Repeating independent samples to match StreamDC's decoded-window count before class balancing does not recover its gains: all four means remain close to Indep. This control tests repeated presentations under the same epoch budget (Appendix C.3).

5.5 Synthesis and Decoding Density

Figure 3 covers 36 HAR–ShapeCond configurations: SPC 1/5/10, $s \in \{L/2, L/4, L/8\}$, and $a \in \{L/2, L/4, L/8, L/16\}$. Reducing a from $L/2$ to $L/16$ raises mean accuracy by 14.24–17.20 points at SPC 1, 2.86–3.30 at SPC 5, and 0.41–0.67 at SPC 10. At SPC 10, $s = L/4$ and $L/8$ peak at $a = L/8$. Denser synthesis offers no consistent improvement; the default $s = L/2$, $a = L/16$ is within 0.27 points of each budget's best mean. Each configuration is synthesized separately, and stride changes readout count, class composition, and potentially optimizer updates under fixed-epoch training. The sweep therefore measures overall pipeline sensitivity (full results and costs: Appendix B).

Table 4: Cross-architecture transfer with ShapeCond and a CNNBN synthesis teacher at equal signal storage. Test accuracy is mean ± standard deviation (%); bold marks the higher mean. *Full (KD)* trains on all real training data with teacher-guided distillation.

		SPC 1		SPC 5		
Dataset	Student	Indep.	StreamDC	Indep.	StreamDC	Full (KD)
HAR	CNNIN	17.95±6.87	**28.04±9.74**	31.01±5.24	**37.31±2.01**	95.89±0.12
	CNNLN	56.24±2.99	**58.84±9.65**	62.42±2.94	**80.97±3.62**	95.92±0.17
	MLP	**62.66±5.38**	61.57±9.19	65.28±4.39	**76.78±2.92**	95.42±0.14
	TCN	22.77±7.46	**53.52±10.26**	64.42±8.62	**81.67±7.73**	95.94±0.10
	GRU	**34.66±4.23**	34.30±6.78	35.90±1.30	**57.44±13.91**	95.89±0.12
	ResNet18BN	28.65±14.99	**54.80±10.97**	65.43±2.85	**83.74±2.65**	95.64±0.42
Sleep	CNNIN	20.72±2.21	**22.68±3.30**	28.10±5.90	**29.57±3.45**	73.52±0.53
	CNNLN	11.26±2.84	**38.97±6.71**	38.35±5.85	**52.82±5.66**	75.75±0.91
	MLP	22.77±15.65	**32.15±7.92**	27.72±14.27	**45.60±1.33**	49.23±2.71
	TCN	**24.46±9.07**	23.89±7.01	15.59±2.82	**38.87±5.13**	72.90±0.55
	GRU	**23.99±10.89**	23.15±5.91	23.76±10.43	**37.69±12.01**	56.45±10.71

5.6 Cross-architecture transfer

With ShapeCond and a CNNBN synthesis teacher, we reuse each method's synthesized data across six student architectures on HAR and five on Sleep (Table 4; protocol in Appendix C.4). StreamDC has higher mean accuracy in 18 of 22 comparisons: all 11 at SPC 5, by 1.47–23.28 points, and seven at SPC 1, with the largest gain of 30.75 points for HAR–TCN. The other four SPC 1 means fall by 0.36–1.09 points (HAR: MLP/GRU; Sleep: TCN/GRU).

6 Conclusion

StreamDC learns a compact temporal memory through jointly optimized linear and class-local cyclic windows. It improves mean accuracy over independent-sequence synthesis in all 30 main configurations at equal signal storage. Controls support the value of adapting shared content during synthesis, while the analysis characterizes gradient coupling and cyclic coverage. StreamDC therefore complements existing condensation objectives.

Future work. Phase-aware readouts may extend StreamDC to tasks where cyclic supervision is unsuitable because labels depend on absolute temporal position. Further work should improve cross-class targets when occupancy misrepresents discriminative evidence, assess teacher reliability on synthetic contexts, and evaluate extensions without teacher relabeling. Finally, reducing the teacher and student cost of dense decoding would complement the savings in signal storage.

AI USE STATEMENT

Generative AI tools assisted with drafting and polishing the manuscript, including improving clarity and checking notation and cross-references. The authors take responsibility for the final content.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide implementation and evaluation details in Appendix C and complete proofs in Appendix A. Source code is submitted as supplementary material for review and will be released publicly upon acceptance.

ETHICS STATEMENT

This work studies data-efficient time-series classification using existing benchmark datasets. It introduces no new data collection or human-subject experiments. Dataset condensation alone does not provide a privacy guarantee.

REFERENCES

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 437–442, 2013. URL https://www.esann.org/sites/default/files/proceedings/legacy/es2013-84.pdf.

Youneng Bao, Yiping Liu, Zhuo Chen, Yongsheng Liang, Mu Li, and Kede Ma. Dataset distillation as data compression: A rate-utility perspective. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 519–529, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Bao_Dataset_Distillation_as_Data_Compression_A_Rate-Utility_Perspective_ICCV_2025_paper.html.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10718–10727, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Cazenavette_Dataset_Distillation_by_Matching_Training_Trajectories_CVPR_2022_paper.html.

Yanping Chen, Adena Why, Gustavo Batista, Agenor Mafra-Neto, and Eamonn Keogh. Flying insect classification with inexpensive sensors. *Journal of Insect Behavior*, 27(5):657–677, 2014. doi: 10.1007/s10905-014-9454-4. URL https://link.springer.com/article/10.1007/s10905-014-9454-4.

Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019. doi: 10.1109/JAS.2019.1911747. URL https://www.ieee-jas.com/article/doi/10.1109/JAS.2019.1911747.

Zhiwei Deng and Olga Russakovsky. Remember the past: Distilling datasets into addressable memories for neural networks. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/de3d2bb604cfc43c81edd2a31b257f03-Abstract-Conference.html.

Jianrong Ding, Zhanyu Liu, Guanjie Zheng, Haiming Jin, and Linghe Kong. CondTSF: One-line plugin of dataset condensation for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e78399fc43dbb2d87b7e1e6906ce5baf-Abstract-Conference.html.

Ary L. Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000. doi: 10.1161/01.CIR.101.23.e215. URL https://doi.org/10.1161/01.CIR.101.23.e215.

Ziyao Guo, Kai Wang, George Cazenavette, Hui Li, Kaipeng Zhang, and Yang You. Towards lossless dataset distillation via difficulty-aligned trajectory matching. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=rTBL8OhdhH.

Pouya Hosseinzadeh, Peiyu Li, Omar Bahri, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. TexTSC: Class-texture preserving data condensation for time series classification. In *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 4222–4230. PMLR, 2026. URL https://proceedings.mlr.press/v300/hosseinzadeh26a.html.

Seoungyoon Kang, Youngsun Lim, and Hyunjung Shim. Label-augmented dataset distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1457–1466, 2025. URL https://openaccess.thecvf.com/content/WACV2025/papers/Kang_Label_Augmented_Dataset_Distillation_WACV_2025_paper.pdf.

Jang-Hyun Kim, Jinuk Kim, Seong Joon Oh, Sangdoo Yun, Hwanjun Song, Joonhyun Jeong, Jung-Woo Ha, and Hyun Oh Song. Dataset condensation via efficient synthetic-data parameterization. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 11102–11118. PMLR, 2022. URL https://proceedings.mlr.press/v162/kim22c.html.

Zhanyu Liu, Ke Hao, Guanjie Zheng, and Yanwei Yu. Dataset condensation for time series classification via dual domain matching. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024. doi: 10.1145/3637528.3671675. URL https://arxiv.org/abs/2403.07245.

Hao Miao, Ziqiao Liu, Yan Zhao, Chenjuan Guo, Bin Yang, Kai Zheng, and Christian S. Jensen. Less is more: Efficient time series dataset condensation via two-fold modal matching. *Proceedings of the VLDB Endowment*, 18(2):226–238, 2024. doi: 10.14778/3705829.3705841. URL https://www.vldb.org/pvldb/vol18/p226-miao.pdf.

Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=l-PrrQrK0QR.

Sijia Peng, Yun Xiong, Xi Chen, Yi Xie, Guanzhi Li, Yanwei Yu, Yangyong Zhu, and Zhiqiang Shen. ShapeCond: Fast shapelet-guided dataset condensation for time series classification. *arXiv preprint arXiv:2602.09008*, 2026. doi: 10.48550/arXiv.2602.09008. URL https://arxiv.org/abs/2602.09008.

Noveen Sachdeva and Julian McAuley. Data distillation: A survey. *Transactions on Machine Learning Research*, 2023. URL https://arxiv.org/abs/2301.04272.

Shitong Shao, Zeyuan Yin, Muxin Zhou, Xindong Zhang, and Zhiqiang Shen. Generalized large-scale data condensation via various backbone and statistical matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/papers/Shao_Generalized_Large-Scale_Data_Condensation_via_Various_Backbone_and_Statistical_Matching_CVPR_2024_paper.pdf.

Zhiqiang Shen, Ammar Sherif, Zeyuan Yin, and Shitong Shao. DELT: A simple diversity-driven EarlyLate training for dataset distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4797–4806, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Shen_DELT_A_Simple_Diversity-driven_EarlyLate_Training_for_Dataset_Distillation_CVPR_2025_paper.html.

Asaf Shul, Eliahu Horwitz, and Yedid Hoshen. Distilling datasets into less than one image. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=qsipSdfWeV.

Julius O. Smith, III. *Spectral Audio Signal Processing*. W3K Publishing, 2011. ISBN 978-0-9745607-3-1. URL https://www.dsprelated.com/freebooks/sasp/Overlap_Add_Decomposition.html.

Peng Sun, Bei Shi, Daiwei Yu, and Tao Lin. On the diversity and realism of distilled dataset: An efficient dataset distillation paradigm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9390–9399, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Sun_On_the_Diversity_and_Realism_of_Distilled_Dataset_An_Efficient_CVPR_2024_paper.html.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. doi: 10.48550/arXiv.1811.10959. URL https://arxiv.org/abs/1811.10959.

Zeyuan Yin, Eric P. Xing, and Zhiqiang Shen. Squeeze, recover and relabel: Dataset condensation at ImageNet scale from a new perspective. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/e91fb65c6324a984ea9ef39a5b84af04-Abstract-Conference.html.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. TS2Vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8980–8987, 2022. doi: 10.1609/aaai.v36i8.20881. URL https://ojs.aaai.org/index.php/AAAI/article/view/20881.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. CutMix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019. URL https://arxiv.org/abs/1905.04899.

Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12674–12685. PMLR, 2021. URL https://proceedings.mlr.press/v139/zhao21a.html.

Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 6514–6523, 2023. URL https://openaccess.thecvf.com/content/WACV2023/html/Zhao_Dataset_Condensation_With_Distribution_Matching_WACV_2023_paper.html.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=mSAKhLYLSsl.

A PROOFS AND SCOPE OF THE ANALYSIS

The statements below concern the readout family specified in Section 3. They neither replace the backend's synthesis loss nor introduce an additional training objective. In particular, the interpolation used in Proposition 2 is a device for analyzing a local perturbation, not a proposed algorithm.

A.1 SELECTION OPERATORS AND ALIGNED COVERAGE

Let $D = dKB$ and $\mathbf{z} = \mathrm{vec}(S) \in \mathbb{R}^D$. For a readout family $\mathcal{V}$ containing n length-L windows, $A_{\mathcal{V}} \in \{0,1\}^{ndL \times D}$ has one unit entry in every row. That entry identifies the stored coordinate selected for the corresponding channel and window position. Cyclic readouts use a modular time index inside their own class segment, without changing the channel index. For $\mathbf{x} = A_{\mathcal{V}}\mathbf{z}$, the chain rule gives $\nabla_{\mathbf{z}} J = A_{\mathcal{V}}^{\top} \nabla_{\mathbf{x}} J$. This uses differentiability of the complete function J, not separability over windows, so batch-statistics matching is allowed. Two distinct columns of $A_{\mathcal{V}}$ cannot have their unit entries in the same row. Consequently the off-diagonal entries of $A_{\mathcal{V}}^{\top} A_{\mathcal{V}}$ vanish, while its ith diagonal entry counts the rows selecting coordinate i. This proves Equation (9). The rank bound follows from the number of columns. The identities describe algebraic coupling and storage dimension; they do not bound the effective sample size or the real-data risk.

Proof of Lemma 1. Write $n = B/s = mq$. Because every class boundary is a multiple of s, the global linear grid restricted to pure windows in any segment has local starts

$$G_{\mathrm{lin}} = \{0, s, \ldots, B - L\}.$$

Its cardinality is $(B - L)/s + 1 = n - q + 1$. The extra-start condition $u + L > B$, applied to $u \in \{0, s, \ldots, B - s\}$, selects exactly

$$G_{\mathrm{extra}} = \{B - L + s, B - L + 2s, \ldots, B - s\}.$$

There are $q - 1$ such starts, and their disjoint union with G_{lin} is the full cyclic grid $G_{\mathrm{cyc}} = \{0, s, \ldots, B - s\}$, of size $n = mq$. Fix a stored time index p on the cycle. The starts of windows containing p are the grid points in the $L = qs$ consecutive integer positions ending at p, interpreted modulo B. Exactly q of those positions are multiples of s. Since $L \le B$, no single window visits p twice. Every stored position therefore has pure-window multiplicity q.

Adjacent linear starts are separated by s when there is more than one; the gap from the last start $B - L$ back to zero around the circle is L. Since $s \le L$, the maximum cyclic gap is L. In the full cyclic grid all gaps are s. This argument also covers the single-start case, whose sole cyclic gap is $B = L$. Substitution of $s = L/2$ gives the count $2m - 1 \to 2m$ and relative increment $1/(2m - 1)$.

The largest-gap result has a limited regularity interpretation. In the discrete geodesic metric on the cycle, every integer offset is within $\lfloor L/2 \rfloor$ of a linear pure start and within $\lfloor s/2 \rfloor$ of a complete cyclic-grid start. A loss that is Λ-Lipschitz in normalized cyclic offset can therefore be compared to its nearest grid loss with errors at most $\Lambda \lfloor L/2 \rfloor / B$ and $\Lambda \lfloor s/2 \rfloor / B$, respectively. These are maximum pointwise interpolation errors under a common loss-regularity assumption. They do not compare the losses achieved by different trained memories.

Nonaligned implementation. For general positive integer stride s, define the exact index sets

$$G_s = \{t \in \mathbb{Z} : 0 \le t \le KB - L,\ t \equiv 0 \pmod{s}\},$$
$$E_s = \{u \in \mathbb{Z} : 0 \le u < B,\ u \equiv 0 \pmod{s},\ u + L > B\}.$$

For segment $j \in \{0, \ldots, K - 1\}$, the local pure starts inherited from the global grid are

$$P_{j,s} = \{t - jB : t \in G_s,\ jB \le t \le (j + 1)B - L\}.$$

Their local phase is $-jB$ modulo s, whereas E_s has phase zero. Thus $P_{j,s} \cup E_s$ need not be the zero-phase cyclic grid; in particular, some local non-tail grid starts can be absent. One must enumerate the actual $P_{j,s}$ and E_s to obtain pure-window counts and coverage multiplicities. Merely replacing the aligned counts with a single per-class floor or ceiling expression can be incorrect. The same distinction applies to decoding with s replaced by a.

A.2 Decoder counting and branchwise approximation

Assume $K \geq 2$, $L \geq 2$, $a \mid L$, and $q = L/a \geq 2$. Since $B = mL$, all segment boundaries align to the stride-a grid. Lemma 1, applied at stride a, gives mq pure cyclic readouts per segment. At an internal boundary $b = jB$, $j \in \{1, \ldots, K-1\}$, the cross-class linear starts are exactly

$$b - L + a,\ b - L + 2a, \ldots, b - a.$$

They number $q - 1$. Since $B \geq L$, a length-L window cannot cross two internal boundaries, and since the global readout is linear, there is no additional last-to-first cross-class boundary. The pure and cross families are disjoint, giving Equation (10). At $a = 1$ the same calculation defines the reference weight ρ_*. For $K = 1$ the cross branch is absent and the claims reduce to the pure branch. At $a = L$ there are no cross-class samples in the coarse decoder; its cross-branch risk can be assigned arbitrarily because its weight is zero. The branchwise regularity formula for a nonempty cross grid below is stated only for $a < L$.

The label rule is part of the loss. Write $r^{\mathrm{p}}_{\theta,j}(u)$ for the actual student loss on class-segment j's cyclic readout at start u, including teacher targets and any backend-specific coefficients. Similarly write $r^{\mathrm{c}}_{\theta,j}(r)$ for the student loss on the readout starting at $jB - L + r$ across internal boundary jB, where $r \in \{1, \ldots, L-1\}$. The teacher-label procedure and any non-offset randomness are fixed consistently across the compared grids. Averaging non-offset randomness is also admissible if the stated bounds hold for the averaged losses. The reference R_* is a finite empirical readout objective on the fixed synthetic memory, not an expectation over real test examples.

Pure branch. For each segment, the stride-a grid contains B/a equally weighted starts ka. Partition all integer offsets into cells $\{ka, ka+1, \ldots, ka+a-1\}$. Couple a grid start ka to a uniformly chosen offset in its cell. The first marginal is the uniform coarse grid and the second marginal is the uniform stride-one grid. If

$$|r^{\mathrm{p}}_{\theta,j}(u) - r^{\mathrm{p}}_{\theta,j}(v)| \leq L_{\mathrm{p}} \frac{d_B(u,v)}{B},$$

where d_B is the cyclic geodesic distance, then the risk discrepancy for that segment is bounded by

$$\frac{L_{\mathrm{p}}}{B} \frac{1}{a} \sum_{v=0}^{a-1} v = \frac{L_{\mathrm{p}}(a-1)}{2B}.$$

Averaging over segments proves the first bound in Equation (13). The same proof works with a within-cell regularity condition instead of global cyclic Lipschitzness.

Cross branch. At a fixed boundary, the coarse offsets are $a, 2a, \ldots, (q-1)a$, while the reference offsets are $1, 2, \ldots, L-1$. Their uniform measures have different endpoint spacing, so the preceding equal-cell proof does not apply directly. Let U be uniform on $(0,1)$ and set

$$X = a\lceil (q-1)U \rceil, \qquad Y = \lceil (L-1)U \rceil.$$

Then X and Y have the required uniform marginals. Conditional on $X = ja$, U lies in $((j-1)/(q-1), j/(q-1)]$ apart from null endpoints. Since $L - 1 = a(q-1) + (a-1)$,

$$Y \geq a(j-1) + \left\lfloor \frac{(j-1)(a-1)}{q-1} \right\rfloor + 1 \geq ja - (a-1),$$

$$Y \leq aj + \left\lceil \frac{j(a-1)}{q-1} \right\rceil \leq ja + (a-1).$$

Hence $|X - Y| \leq a - 1$ almost surely. Under

$$|r^{\mathrm{c}}_{\theta,j}(r) - r^{\mathrm{c}}_{\theta,j}(r')| \leq L_{\mathrm{c}} \frac{|r - r'|}{L},$$

the corresponding loss difference is at most $L_{\mathrm{c}}(a-1)/L$ in expectation. Averaging over the $K-1$ boundaries gives the cross-branch bound. Together, the two branches satisfy

$$\epsilon_{\mathrm{p}} \leq \frac{L_{\mathrm{p}}(a-1)}{2B}, \qquad \epsilon_{\mathrm{c}} \leq \frac{L_{\mathrm{c}}(a-1)}{L}. \tag{13}$$

The constants must cover the labels as well as the predictions. A hard teacher-label switch, or a switch in the dominant-class target when $\beta_b > 0$ in Equation (8), can make offset regularity poor; the condition does not follow merely from smooth stored signals.

A.3 Risk decomposition and optimizer transfer

Proposition 1 (Readout risk decomposition)**.** *Fix the synthetic memory and label rule. Let R_a and R_* be the uniform stride-a and stride-one readout risks defined in Section 4, and assume all considered per-window student losses lie in $[0, M]$. If $|R_{\mathrm{p},a}(\theta) - R_{\mathrm{p},*}(\theta)| \le \epsilon_{\mathrm{p}}$ and $|R_{\mathrm{c},a}(\theta) - R_{\mathrm{c},*}(\theta)| \le \epsilon_{\mathrm{c}}$, then Equation* (11) *holds. Reweighting the decoded examples adds at most $M\,\mathrm{TV}(Q_a, P_a)$, where P_a is the uniform decoder measure and Q_a is the resulting sampling measure. Let F_a be the objective actually minimized, either R_a or its reweighted version. If the resulting bound $\sup_\theta |F_a(\theta) - R_*(\theta)| \le \epsilon_a$ holds and $F_a(\hat\theta_a) \le \inf_\theta F_a(\theta) + \eta_{\mathrm{opt}}$, then*

$$R_*(\hat\theta_a) - \inf_\theta R_*(\theta) \le 2\epsilon_a + \eta_{\mathrm{opt}}. \tag{14}$$

Proof of Proposition 1. Add and subtract $(1-\rho_a)R_{\mathrm{p},*} + \rho_a R_{\mathrm{c},*}$ to obtain

$$\begin{aligned} R_a - R_* = (1-\rho_a)(R_{\mathrm{p},a} - R_{\mathrm{p},*}) + \rho_a(R_{\mathrm{c},a} - R_{\mathrm{c},*}) \\ + (\rho_a - \rho_*)(R_{\mathrm{c},*} - R_{\mathrm{p},*}). \end{aligned}$$

Both reference branch risks lie in $[0, M]$, so their difference has absolute value at most M. Applying the triangle inequality establishes Equation (11).

View every decoded window, with its readout type and fixed label rule, as an atom in the stride-one index space. A global cross-class readout and a local cyclic readout can share a numerical start and remain distinct atoms. Under aligned grids the coarse atoms are a subset of that space. If oversampling or weighting changes P_a to Q_a, use the convention

$$\mathrm{TV}(Q_a, P_a) = \frac{1}{2}\sum_v |Q_a(v) - P_a(v)|.$$

For any function r with values in $[0, M]$,

$$\left|\sum_v r(v)\big(Q_a(v) - P_a(v)\big)\right| \le M\,\mathrm{TV}(Q_a, P_a).$$

This follows by splitting the signed measure into positive and negative parts, each of mass $\mathrm{TV}(Q_a, P_a)$. It proves the stated additional reweighting term. Nonuniform replication or class-dependent resampling can change this measure without creating new readout coordinates; uniform replication of every atom leaves the normalized measure unchanged. This argument describes the expected sampling objective and does not itself analyze minibatch noise or coupled training-time batch statistics in the student.

Let F_a denote the objective actually minimized: R_a for uniform decoding, or its reweighted version otherwise. Assume $\sup_\theta |F_a(\theta) - R_*(\theta)| \le \epsilon_a$ and $F_a(\hat\theta_a) \le \inf_\theta F_a(\theta) + \eta_{\mathrm{opt}}$. Then

$$\begin{aligned} R_*(\hat\theta_a) &\le F_a(\hat\theta_a) + \epsilon_a \\ &\le \inf_\theta F_a(\theta) + \eta_{\mathrm{opt}} + \epsilon_a \\ &\le \inf_\theta R_*(\theta) + \eta_{\mathrm{opt}} + 2\epsilon_a, \end{aligned}$$

which is Equation (14). No existence of a global minimizer of R_* is needed for this argument.

The uniform premise is stronger than a bound evaluated at one fixed model. Finite observed loss variation alone does not establish it over all neural-network parameters. One can instead state the result on an explicit restricted class with bounded logits and uniform offset regularity. For CE and KL, finite positive temperatures and bounded teacher and student logits suffice for a finite loss bound; for MSE, bounded compared logits suffice. A combined backend loss needs these conditions for all its terms and fixed finite coefficients. No monotonic improvement of observed risk or test accuracy follows: the inequalities are upper bounds, the learned optimizer can have nonzero error, and the synthetic reference risk may differ from the real-data risk.

A.4 LOCAL EFFECT OF JOINT SYNTHESIS

Proposition 2 (Local compatibility condition)**.** *Fix teacher states, readout sets, and stochastic choices. Let J_0 be the complete linear-only synthesis objective and $C = J_{\mathrm{hybrid}} - J_0$. Suppose J_0 and C are C^3 near a stationary point S_0, with $H = \nabla^2 J_0(S_0) \succ 0$ in vectorized coordinates. Let R_{d} be a C^2 held-out-readout risk for a fixed evaluation model and locally regular label rule. For the local minimizer S_λ of $J_\lambda = J_0 + \lambda C$ continuing from S_0, Equation* (12) *holds as $\lambda \to 0$.*

Proof of Proposition 2. Work in vectorized coordinates $\mathbf{z} = \mathrm{vec}(S)$ and write

$$F(\mathbf{z}, \lambda) = \nabla J_0(\mathbf{z}) + \lambda \nabla C(\mathbf{z}).$$

The C^3 assumptions make F a C^2 map near $(\mathbf{z}_0, 0)$. At that point $F(\mathbf{z}_0, 0) = 0$ and its derivative in $\mathbf{z}$ is the invertible matrix H. The implicit function theorem yields a locally unique C^2 stationary branch $\mathbf{z}_\lambda$. By continuity, its Hessian remains positive definite for sufficiently small $|\lambda|$, so this branch consists of strict local minimizers. Differentiating $F(\mathbf{z}_\lambda, \lambda) = 0$ at zero gives

$$H \left.\frac{d\mathbf{z}_\lambda}{d\lambda}\right|_{\lambda=0} + \nabla C(\mathbf{z}_0) = 0.$$

Taylor expansion therefore gives

$$\mathbf{z}_\lambda = \mathbf{z}_0 - \lambda H^{-1} \nabla C(\mathbf{z}_0) + O(\lambda^2).$$

Expanding the C^2 function R_{d} around $\mathbf{z}_0$ establishes Equation (12). If its displayed inner product is strictly positive, the negative linear term dominates the quadratic remainder for all sufficiently small positive λ.

This proposition accommodates batch-coupled synthesis terms because C is the difference of complete objectives. It does not assume that the derivative equals a sum of cyclic-window gradients. In the actual pipelines, adding views can also change CE normalization, matched batch statistics, and the availability of within-class diversity pairs; all such effects belong to C. For nonsmooth networks or label rules, the smoothness premise is only valid within an appropriate stable smooth region, if one exists. A finite synthesis checkpoint need not be stationary, and a flat or singular Hessian does not satisfy the stated hypothesis. These are substantive conditions rather than properties established for the reported implementations. Ordinary gradient alignment is only a diagnostic proxy for the Hessian-weighted compatibility in the formula. Finally, the proposition fixes the evaluation model. An analysis of full downstream training as a function of S would require additional bilevel regularity and stability conditions and is not claimed here.

A.5 A DEGENERACY UNDER EXACT CYCLIC INVARIANCE

Consider $m = 1$, so a class segment is a single stored length-L sample. Let T_u be the permutation matrix for a cyclic shift by u, and let $g_c(\mathbf{z}) = \ell(f(\mathbf{z}), e_c)$ be a differentiable, additive per-window synthesis loss for that class. Suppose that, for the shifts under consideration,

$$g_c(T_u \mathbf{z}) = g_c(\mathbf{z})$$

throughout a neighborhood of the stored sample. This follows, for example, if the predictions of f are exactly cyclically invariant there and the class target is fixed. Each cyclic-window loss then equals the loss on a duplicate of the original pure sample. Differentiating the identity also gives

$$T_u^\top \nabla g_c(T_u \mathbf{z}) = \nabla g_c(\mathbf{z}).$$

Thus the gradients with respect to the stored sample are identical, including the readout pullback. With the same number and weights of added pure views, cyclic supervision and duplicate-pure supervision have the same additive objective and gradient in this setting.

The conclusion concerns the specified additive component only. Batch-statistics and diversity terms can distinguish the two view families even if a teacher's final predictions do not. It also requires functional invariance in a neighborhood; equal predictions at a single sample are insufficient to establish gradient equality. For $m > 1$, windows can select different subsequences of a longer segment and are not generally cyclic shifts of one fixed sample, so the premise must not be extended without further argument.

B COMPLETE SYNTHESIS–DECODING DENSITY RESULTS

Table 5 reports the complete accuracy grid on HAR with ShapeCond. Table 6 reports the corresponding synthesis costs, decoded-window counts, and student-stage costs.

Evaluation and measurement protocol. Every (m, s, a) configuration is executed separately, using three teacher-initialization seeds and three synthesis repetitions per seed. Student checkpoints are selected by minimum cross-entropy on a separate validation split (Appendix C.2); their final test accuracies are summarized across these nine runs. Synthesized memories are not reused across decoding-stride settings. Student training uses a fixed number of epochs, so changing the decoded set can also change the number of optimizer updates. Measurements use an NVIDIA RTX 4090 GPU. During student training, each minibatch is passed through the teacher to produce the soft targets for the KL loss. The recorded student-stage time spans model construction, training with these online teacher forward passes, validation at each epoch, periodic test evaluations, checkpoint saving and loading, and final evaluation. There is no separate relabeling stage in this pipeline. Reported times are means over the nine runs. Peak GPU memory is the peak allocation recorded by the process's PyTorch allocator, and the table reports the maximum of these per-run peaks.

Table 5: Complete synthesis–decoding stride grid on HAR with ShapeCond. Test accuracy is reported as mean ± standard deviation (%). Each configuration aggregates three teacher-initialization seeds and three synthesis repetitions per seed. Bold identifies the highest mean within each SPC budget. The default setting is $s = L/2$, $a = L/16$.

		Decoding stride a			
SPC	Synthesis stride s	$L/2$	$L/4$	$L/8$	$L/16$
1	$L/2$	72.11±4.04	78.15±3.01	84.44±1.95	86.35±2.10
	$L/4$	69.47±2.29	77.29±3.57	84.44±2.72	**86.62±2.10**
	$L/8$	67.83±3.78	76.77±2.56	81.98±3.65	85.03±2.71
5	$L/2$	91.06±0.89	91.68±0.72	93.67±0.31	93.92±0.65
	$L/4$	91.24±0.79	91.72±0.75	93.35±0.51	**94.19±0.82**
	$L/8$	90.43±1.13	91.80±0.82	93.64±0.75	93.73±1.21
10	$L/2$	93.50±0.46	93.70±0.82	94.02±0.62	94.17±0.54
	$L/4$	93.47±0.54	93.96±0.37	**94.18±0.90**	93.88±0.60
	$L/8$	93.10±0.58	93.60±0.40	93.86±0.62	93.67±0.44

Readout counts and composition. HAR has $K = 6$ classes and sequence length $L = 128$. All tested strides divide L, so the aligned counting formulas apply. With $q = L/a$, the decoder contains $N_{\mathrm{p}} = 6mq$ pure-class windows and $N_{\mathrm{c}} = 5(q-1)$ cross-class windows, giving $N = (6m+5)q-5$ windows in total. These counts match every recorded configuration. Reducing a from $L/2$ to $L/16$ increases N from 17 to 171 at SPC 1, from 65 to 555 at SPC 5, and from 125 to 1035 at SPC 10. Under uniform weighting of these readouts, the cross-class fraction changes from 29.41% to 43.86%, from 7.69% to 13.51%, and from 4.00% to 7.25%, respectively. The sweep characterizes the overall sensitivity of the pipeline to stride choices, including changes in readout counts and composition under fixed-epoch training; it does not isolate the temporal-resolution term alone.

Recorded costs. With $s = L/2$, reducing a from $L/2$ to $L/16$ changes the recorded student-stage time from 2.5 to 2.7 minutes at SPC 1, from 2.4 to 3.2 minutes at SPC 5, and from 2.5 to 4.0 minutes at SPC 10. At $a = L/16$, increasing synthesis density from $s = L/2$ to $s = L/8$ raises recorded synthesis peak memory from 123 to 150 MiB, from 147 to 237 MiB, and from 177 to 340 MiB for the three budgets, without a consistent accuracy improvement. These are measured costs for the reported configurations.

Table 6: Recorded costs for the complete HAR stride grid. N is the decoded-window count before class balancing. Measurements use an NVIDIA RTX 4090 GPU. Student-stage time includes training, online teacher inference, evaluation, and checkpoint handling. Times are means over nine runs; peak memory is the maximum of the nine process-level PyTorch allocation peaks. Times are in minutes and memory is in MiB (2^{20} bytes).

				Synthesis		Student stage	
SPC	s	a	N	Time (min)	Peak (MiB)	Time (min)	Peak (MiB)
1	$L/2$	$L/2$	17	0.6	122	2.5	118
		$L/4$	39	0.5	122	2.4	118
		$L/8$	83	0.6	122	2.5	119
		$L/16$	171	0.5	123	2.7	120
	$L/4$	$L/2$	17	0.6	130	2.4	118
		$L/4$	39	0.6	131	2.4	119
		$L/8$	83	0.6	131	2.5	119
		$L/16$	171	0.6	132	2.7	120
	$L/8$	$L/2$	17	0.6	148	2.3	119
		$L/4$	39	0.6	148	2.4	119
		$L/8$	83	0.6	149	2.4	119
		$L/16$	171	0.6	150	2.6	120
5	$L/2$	$L/2$	65	0.6	142	2.4	119
		$L/4$	135	0.6	142	2.6	120
		$L/8$	275	0.6	144	2.8	122
		$L/16$	555	0.6	147	3.2	124
	$L/4$	$L/2$	65	0.6	171	2.4	120
		$L/4$	135	0.6	172	2.6	120
		$L/8$	275	0.6	174	2.7	122
		$L/16$	555	0.6	177	3.2	124
	$L/8$	$L/2$	65	0.7	230	2.4	120
		$L/4$	135	0.7	231	2.5	121
		$L/8$	275	0.7	232	2.7	122
		$L/16$	555	0.7	237	3.1	126
10	$L/2$	$L/2$	125	0.6	167	2.5	121
		$L/4$	255	0.6	170	2.7	122
		$L/8$	515	0.6	172	3.1	125
		$L/16$	1035	0.6	177	4.0	130
	$L/4$	$L/2$	125	0.7	223	2.5	121
		$L/4$	255	0.7	224	2.7	123
		$L/8$	515	0.8	226	3.3	125
		$L/16$	1035	0.8	233	4.2	130
	$L/8$	$L/2$	125	1.0	331	2.8	122
		$L/4$	255	1.0	332	3.0	124
		$L/8$	515	0.9	333	3.1	127
		$L/16$	1035	1.0	340	4.4	131

C Implementation and evaluation details

C.1 Readout design and relation to shared representations

PoDD and StreamDC both replace independent synthetic examples with a shared stored representation accessed by overlapping readouts. PoDD organizes image content on a two-dimensional poster and extracts rectangular patches, using majority-based fixed targets or a learned spatial label field (Shul et al., 2025).

StreamDC's readout structure addresses temporal placement under a fixed signal storage budget. A one-dimensional memory concatenates class-specific segments. Global linear windows expose contiguous temporal contexts, including cross-class compositions, while class-local cyclic windows join each segment's endpoint portions. These cyclic reads change the adjacency at a class-local seam; refining the grid of contained linear windows cannot produce them. At $m = 1$, a length-L class segment admits only one contained length-L window, whereas cyclic completion supplies additional pure-class temporal origins. Under aligned grids, the construction also gives uniform pure-class coverage (Lemma 1).

Boundary supervision and the two processing stages are specified separately. Cross-class linear windows receive temporal-occupancy targets during synthesis, while cyclic windows retain their segment's class. Both families jointly optimize the same memory coordinates. After synthesis, the memory is frozen and read at an independently chosen decoding stride, with teacher predictions supplying downstream targets. This separates the density of contexts used to optimize the memory from the density exposed to student training.

IDC generates views by partitioning and resizing compact inputs, including time-axis formation on speech spectrograms (Kim et al., 2022); StreamDC changes temporal origins while preserving the spacing of the selected signal coordinates. Appendix D shows examples of the learned temporal memories.

C.2 Backend objectives and evaluation protocol

Synthesis objectives. The SRe^{2}L-style and ShapeCond adaptations use $0.99\mathcal{L}_{\mathrm{cls}} + 0.01\mathcal{R}_{\mathrm{BN}}$, where $\mathcal{R}_{\mathrm{BN}}$ matches the batch-normalization input means and variances to the teacher's stored statistics. The first BN layer has weight 10 and the remaining layers have weight 1. SRe^{2}L-style uses a plain CNNBN teacher; ShapeCond uses its shapelet-assisted counterpart. The independent baseline synthesizes groups of K independent windows, whereas StreamDC optimizes the shared memory jointly over its readout family. Both use the same backend loss components and coefficients.

The G-VBSM adaptation rotates CNNBN, CNNLN, and MLP teachers during synthesis and uses $\mathcal{L}_{\mathrm{cls}} + 0.05\mathcal{R}_{\mathrm{stat}} + 10^{-5}\mathcal{L}_{\mathrm{div}}$. Here $\mathcal{R}_{\mathrm{stat}}$ is the generalized and moving-average statistics-matching term, with first-layer multiplier 10, statistics momentum 0.8, and hook drop probability 0.4. $\mathcal{L}_{\mathrm{div}}$ is the within-class Gram-spectrum diversity term, using temperature 4 and temporal pooling to length 32. For mixed windows, its class grouping uses the dominant occupancy class. A class represented by a single window contributes zero to this diversity term. The generalized statistics are collected from convolutional inputs on the real training data, including channel and four-position patch means and variances. Synthetic statistics use the moving-average estimates during optimization. G-VBSM clips synthetic values to the per-channel training-data range after each update. These are the time-series backend implementations used for both Indep. and StreamDC.

Student objectives. For SRe^{2}L-style and ShapeCond, define $p_T(x) = \mathrm{softmax}(f_T(x)/T)$ and $p_\theta(x) = \mathrm{softmax}(f_\theta(x)/T)$. The student minimizes

$$\ell^{\mathrm{stu}}(x) = \mathrm{KL}(p_T(x) \,\|\, p_\theta(x)), \qquad T = 4, \quad \beta_b = 0. \tag{15}$$

The loss is averaged over examples without an additional T^2 factor. Teacher probabilities are computed online for each minibatch. The default ShapeCond CNNBN student also retains the shapelet feature branch. For G-VBSM, let $\bar{z}_T(x)$ be the mean logits of the three teachers. These targets are computed for the synthesized examples and replayed during student training, which minimizes

$$\ell^{\mathrm{stu}}_{\text{G-VBSM}}(x) = \frac{1}{K}\|f_\theta(x) - \bar{z}_T(x)\|_2^2 + 0.01\,\mathrm{CE}\big(f_\theta(x), y^{\mathrm{dom}}\big). \tag{16}$$

Data sources. We use the HAR, Electric (ElectricDevices), Sleep, and Insect (InsectSound) data packages distributed with CondTSC, and the FacesUCR package obtained through ShapeCond. FacesUCR and ElectricDevices belong to the UCR archive (Dau et al., 2019); the other source references follow CondTSC (Anguita et al., 2013; Goldberger et al., 2000; Chen et al., 2014).

Data preparation and model selection. We use the classwise partitions produced by the data-preparation pipeline. For HAR, Electric, Sleep, and Insect, the source partitions are pooled and repartitioned within each class using the source split size ratios; FacesUCR uses a classwise 60%/20%/20% partition after pooling its source training and test sets. The training partition is class-balanced by random oversampling. Channel normalization statistics are computed on the prepared training partition and reused for the validation, test, and synthesized data.

Teacher and student checkpoints are selected by minimum cross-entropy on the validation split. The selected checkpoint is reloaded and evaluated on the separate test split. Dataset sizes refer to the partitions before oversampling.

Optimization settings. Tables 7 and 8 give the data dimensions, default strides, and dataset-specific settings. Teacher and student networks use AdamW with $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and weight decay 10^{-4}. Their training batch sizes are 256 and 20, respectively; evaluation uses batch size 20, except for Insect, which uses 256. Signal synthesis uses Adam with $(\beta_1, \beta_2) = (0.5, 0.9)$, $\epsilon = 10^{-8}$, and zero weight decay. All three stages use cosine learning-rate decay with no warm-up. Each synthesis step evaluates the complete window group being optimized: K windows per independent group for SRe^{2}L-style and ShapeCond, all Km independent windows jointly for G-VBSM, or all synthesis readouts for the shared memory. We retain the memory snapshot with the lowest observed synthesis objective. The Insect experiments reuse the existing teacher checkpoints while using synthesis learning rate 0.02 and student learning rate 2×10^{-4} for 200 epochs. Teachers are frozen in evaluation mode during synthesis and student training. The reported runs use raw inputs and one optimizer update per minibatch.

Shapelet features and cached teachers. ShapeCond reuses the training-data shapelet cache across synthesis repetitions, loading 10 shapelets per channel, or 20 for Sleep. For each shapelet, the differentiable feature is its minimum Euclidean distance to same-length subsequences in the corresponding input channel, evaluated at unit stride. Shapelet features are concatenated to the learned representation in the supported architectures (Appendix C.5). Pretrained teacher checkpoints are reused across representations, SPC budgets, and stride settings for a given teacher initialization. Outer seed 2025 uses the existing teacher checkpoints; the remaining seeds are pretrained with the configured settings before the experiment grids. The cached shapelets and teacher states are required to reproduce these backend protocols; the dKB signal budget counts the condensed signal itself.

Randomness and aggregation. The main comparisons use network-initialization seeds $\{2025, 2026, 2027\}$, with three synthesis repetitions for each teacher initialization. Each outer process initializes its Python, NumPy, and PyTorch random streams with seed 2025; network construction then uses the designated outer seed. The three synthesis repetitions run sequentially, and their training-example samples advance the process's random stream. The student construction uses the same outer seed. Mean and standard deviation are computed over the nine resulting test accuracies, using the population standard deviation (denominator 9). Class balancing uses random oversampling with a fixed resampling seed of 0 and balances each class to the largest class count in that loader.

Experimental variants and hardware. The controls on HAR and Sleep inherit their corresponding ShapeCond settings in Table 8. The HAR density study changes only the SPC budget and the synthesis/decoding strides specified in Appendix B. The transfer study uses the HAR and Sleep student learning rates and epoch budgets from the same table, with its repetition structure specified in Appendix C.4. All reported experiments run on an NVIDIA RTX 4090 GPU. Signal values and network computations use 32-bit floating point. The resource measurements and their aggregation are described in Appendix B.

Table 7: Dataset profiles and default readout strides. Partition sizes are counted before class balancing. Validation is used for checkpoint selection and test for reported accuracy. K, d, and L denote classes, channels, and sequence length; s and a are the default synthesis and decoding strides for StreamDC.

Dataset	Train	Validation	Test	K	d	L	s	a
HAR	5878	2944	1477	6	9	128	64	8
FacesUCR	1344	449	457	14	1	131	65	8
Electric	7807	7709	1121	7	1	96	48	6
Sleep	25609	8908	7791	5	1	3000	1500	187
Insect	21870	25000	3130	10	1	600	300	37

Table 8: Dataset-dependent optimization settings used in the main comparisons for both representations and all three backends. Learning rates are initial values before cosine decay. Each optimized independent batch or shared memory receives 2,000 synthesis iterations. The final column is the number of shapelets per channel used by ShapeCond.

	Teacher		Synthesis	Student		Shapelets
Dataset	LR	Epochs	LR	LR	Epochs	per channel
HAR	10^{-4}	150	0.20	10^{-3}	150	10
FacesUCR	10^{-4}	150	0.20	10^{-3}	150	10
Electric	10^{-4}	100	0.35	5×10^{-4}	100	10
Sleep	10^{-5}	150	0.15	2×10^{-4}	100	20
Insect	10^{-4}	150	0.02	2×10^{-4}	200	10

C.3 Window variants and replication control

Window-family variants. The variant without cyclic windows uses the global linear grid alone at both synthesis stride s and decoding stride a. The variant without cross-class windows uses a linear grid anchored at the beginning of each class segment, together with that segment's cyclic extras, at both stages. At stride $h \in \{s, a\}$ its local linear starts are $0, h, 2h, \ldots \leq B - L$, and its cyclic extras are the starts $u = 0, h, 2h, \ldots < B$ satisfying $u + L > B$. This construction keeps all windows within their assigned class segment. If h does not divide B, anchoring the linear grid separately in each segment can also change its pure-window offsets relative to the global grid. This applies to Sleep decoding ($L = 3000$, $a = 187$); the HAR grids ($L = 128$, $s = 64$, $a = 8$) are aligned.

Replication before class balancing. The replication control first repeats independently synthesized samples to match StreamDC's total decoded-window count before class balancing. It then applies the same class-balancing loader and student-training protocol. Since the class counts before balancing differ, the resulting training-set sizes need not coincide. Table 9 reports the resulting sizes and optimizer updates per epoch, verified across the nine runs of each setting. This control evaluates the tested replication scheme under the same epoch budget; it does not impose an identical number of optimizer updates in every setting.

C.4 Cross-architecture evaluation and full-data reference

For each method, SPC budget, and teacher-initialization seed, we reuse one stored synthesized dataset across all student architectures. Each student architecture is trained with initialization seeds $\{101, 102, 103\}$ on each of the three stored datasets, giving nine evaluations per Indep.–StreamDC configuration. The CNNBN synthesis teacher associated with each stored dataset supplies the online KL targets.

The *Full (KD)* column uses the complete real training partition as student input while retaining the same teacher-guided KL objective and the corresponding student optimization and model-selection protocol. It reports three student initializations with the fixed CNNBN teacher from outer seed 2025. The full-data loader uses batch size 256, while the synthesized-data loader uses batch size 20. Thus

Table 9: Counts for the replication control. N is the common sample count before class balancing. The remaining columns give the actual training-set sizes after balancing and optimizer updates per epoch with batch size 20.

			After balancing		Updates/epoch	
Dataset	SPC	N	StreamDC	Replicated	StreamDC	Replicated
HAR	1	171	192	174	10	9
HAR	5	555	576	558	29	28
Sleep	1	145	160	145	8	8
Sleep	5	466	480	470	24	24

Table 10: Architectures used in the main and transfer experiments. Convolutional kernels and pooling operate along time. BN, IN, and LN denote batch, instance, and layer normalization, respectively.

Network	Configuration
CNNBN/IN/LN	Three blocks of width-32 convolution (kernel 3, stride 1, padding 1), normalization, ReLU, and max pooling (kernel/stride 2), followed by flattening and a linear classifier. IN is affine per channel; LN covers the channel–time feature shape. ShapeCond CNNBN concatenates dn_s shapelet features before its classifier.
MLP	Flattened input; hidden widths 128 and 128, each with BN and ReLU, then a linear classifier. ShapeCond uses second hidden width $128 - dn_s$ and concatenates the shapelet features to restore a 128-dimensional classifier input.
TCN	One residual block with two weight-normalized causal convolutions (width 64, kernel 7, dilation 1), ReLU, zero dropout, and a 1×1 residual projection; temporal global average pooling and a linear classifier.
GRU	One unidirectional layer with hidden width 100 and no dropout; the final time-step output feeds a linear classifier.
ResNet18BN	HAR only. Width-64 convolutional stem with BN and ReLU; four stages with 2/2/2/2 basic blocks and widths 64/128/256/512, temporal kernel 3, and stage strides 1/2/2/2. Fixed temporal average pooling of width 4 precedes the 2,048-dimensional linear head.

it is a full-data distillation reference. Its values describe the same supervision protocol applied to real training data, and are not a performance bound.

C.5 Network architectures

Table 10 summarizes the network definitions used in the experiments. CNNBN is the default student for the main comparisons. The SRe^{2}L-style teacher is plain CNNBN, the ShapeCond teacher is its shapelet-assisted counterpart, and G-VBSM uses the plain CNNBN, CNNLN, and MLP teacher collection. In ShapeCond, the CNNBN and MLP implementations concatenate shapelet features; the CNNIN, CNNLN, TCN, GRU, and ResNet18BN implementations use their listed network features. If n_s shapelets are used per channel, the additional feature vector has dimension dn_s.

C.6 Existing-method comparison protocols

Table 2 uses the same prepared data splits, with three repeated runs per baseline. Each method retains its own architecture, condensation objective, augmentation, and student-training protocol. This comparison evaluates complete pipelines; Table 1 separately isolates StreamDC within each backend.

Full, Random, and K-center. These references use the shapelet-assisted CNNBN student with hard-label cross-entropy. Random samples m training examples per class uniformly without replacement. K-center selects within each class using Euclidean distance between normalized, flattened signals: it starts at the example closest to the class mean and repeatedly chooses the point farthest from its nearest selected center. Student optimization follows the ShapeCond settings, except that these Insect references use learning rate 10^{-3} and 150 epochs. Evaluation batch size is 256.

Entries summarize seeds $\{2025, 2026, 2027\}$ with population standard deviation. The Full column in Table 1 uses hard-label supervision; Full (KD) in Table 4 retains its separate distillation protocol.

CondTSC and TexTSC. We reproduce CondTSC and TexTSC using their released implementations, adapting data loading to the shared splits. CondTSC retains dual-domain matching with CNNBN, expert trajectories, and its evaluation procedure. TexTSC retains its TinyClassifier, K-means initialization, statistical and gradient-matching objectives, and training augmentation. TexTSC's teacher guides synthesis; its fresh downstream student uses hard-label cross-entropy rather than StreamDC's per-window teacher targets.

D VISUALIZATIONS OF SYNTHESIZED MEMORIES

Figure 4 shows examples of the class-segmented memories learned by StreamDC with ShapeCond on the five datasets at SPC 1. Each segment stores L time steps, and the full memory contains KL time steps per channel. The displayed class order is fixed during synthesis.

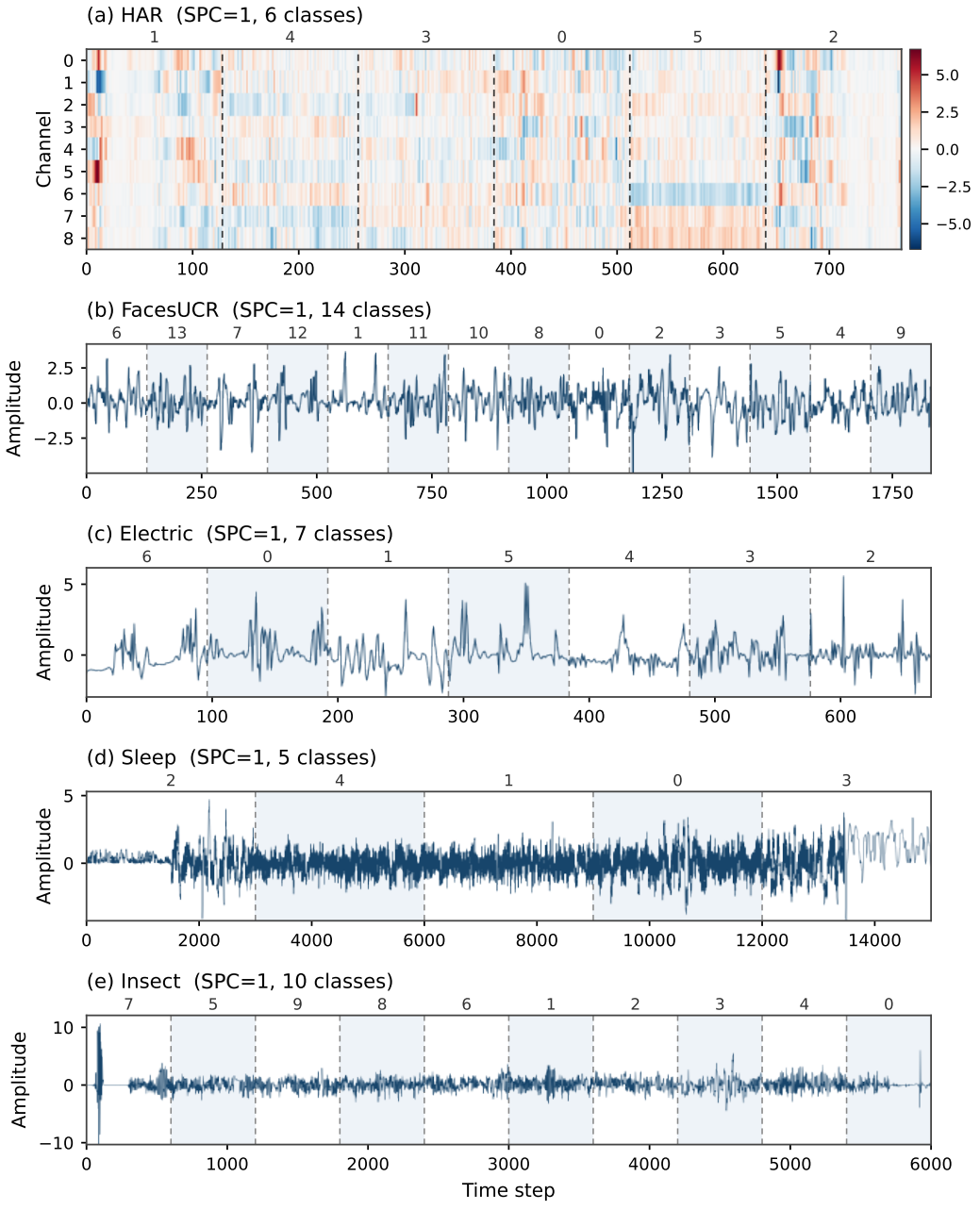

Figure 4: **Examples of synthesized temporal memories at SPC 1.** HAR is shown as a channel–time heatmap; FacesUCR, Electric, Sleep, and Insect are shown as single-channel amplitude traces. Dashed vertical lines mark class-segment boundaries, and the numbers above the segments identify their classes. The HAR color scale represents signal amplitude. These plots show the stored memories before window decoding.